MIXED SEMIDEFINITE AND SECOND-ORDER CONE OPTIMIZATION APPROACH FOR THE HANKEL MATRIX APPROXIMATION PROBLEM

By
Mohammed M Alshahrani

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

In Partial Fullfillment of the Requirements
for the degree
Master of Science
IN
Mathematics

KING FAHD UNIVERSITY
OF PETROLEUM & MINERALS
Dhahran, Saudi Arabia
April, 2003

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

«وَقُلْ رَبِّي زِدْنِي عِلْمًا»

KING FAHD UNVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Mohammed M Alshahrani** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements of the degree of **MASTER OF SCIENCE** in Mathematics.

Thesis Committee

Dr. Suliman Al-Homidan, Thesis Advisor

Dr. Basem Attili, Member

Dr. Mohamed Bendaya, Member

Department Chairman

Dean of Graduate Studies

Date

Dedication

I wish to dedicate this thesis to my parents,
my brothers, my sisters, my wife, my kids and
my freind Mohammed Naser

Acknowledgment

First of all I thank "ALLAH" The Almighty for the knowledge, help and guidance HE has showered on me. Thanks are also to our Prophet Mohammad (pbuh) who encouraged us as Muslims to seek knowledge and stressed that science and Islam are never separable.

I am greatly indebted to my thesis advisor, Dr. Suliman Al-Homidan, for his guidance, help, support, patience and encouragement in making this work possible. I am sincerely grateful to my thesis members, Dr. Basem Attili and Professor Mohamed Benday for their helpful comments.

Acknowledgments are due to King Fahd University of Petroleum and Minerals for giving me the opportunity to pursue my study. I also acknowledge Teachers' College in Dammam for supporting and encouragement. A special thank is due to Dr. Ahmad Alfkhalaf, chairman of the department of mathematics at Dammam Teachers' college, and my colleagues especially Dr. Zakariya for their support and help.

My sincere appreciation are due to Dr. Salim Masoudi, my academic advisor, Dr. Saleh Eddine Kabbaj, my example of endurance and hard work, Dr. Waleed Alsbah, ex-chairman, Dr. Khalid Furati, chairman, and all other faculty of the department of mathematical sciences for their concern and encouragement.

Last but not least, I put in record my high appreciation to the support of my wife and kids during the study period.

Contents

List of Tables

List of Figures

Thesis Abstract

Name	:	Mohammed M Alshahrani
Title of thesis	:	Mixed semidefinite and second-order cone optimization approach for the Hankel matrix approximation problem
Major	:	Mathematics (Optimization)
Date of degree	:	April, 2003

Approximating the nearest positive semidefinite Hankel matrix in the Frobenius norm to an arbitrary data covariance matrix is useful in many areas of engineering, including signal processing and control theory. In this thesis, the powerful interior point primal-dual path-following method will be used to solve our problem after reformulating it into different forms, first as a semidefinite programming problem, then into the form of a mixed semidefintie and second-order cone optimization problem. Numerical results, comparing the performance of these methods against the modified alternating projection method will be reported.

ملخص الرسالة

الاســـــــــم : محمّد معجب الشهراني.

عنوان الرسالة : حل مسألة تقريب مصفوفة «هانكل » كمسألة أمثلية مختلطة ذات قيود في المخروط شبه المحدد والمخروط ذي الرتبة الثانية.

التخـصـــص : ريّاضيّات «أمثلية»

تاريخ التخرج : صفر ، ١٤٢٤ هـ (أبريل، ٢٠٠٣ م)

إن إيجاد أقرب مصفوفة «هانكل» شبه محددة من مصفوفة أخرى معطاة مفيد في كثير من التطبيقات الهندسية، بما في ذلك معالجة الإشارات و نظرية التحكم. في هذا البحث سنستخدم طريقة النقطة الداخلية التابعة للمسار لحل المسألة بعد إعادة صياغتها بطرق مختلفة، أولا كمسألة برمجة شبه محددة، ثمّ كمسألة أمثلية مختلطة شبه محددة و ثنائية الرتبة. هذا البحث يعطي نتائج عددية تقارن أداء هذه الطريقة «التابعة للمسار» بصياغاتها المختلفة مع طريقة الإسقاط المتناوب.

Chapter 1

Introduction

1.1 Preamble

In some application areas, such as digital signal processing and control theory, it is required to compute the closest, in some sense, positive semidefinite Hankel matrix, with no restriction on its rank, to a given data covariance matrix, computed from a data sequence. In signal processing, for example, the Hankel matrix (and other structured matrices) approximation problem is of practical interest, where replacing a data matrix corresponding to a signal plus a noise with a matrix known to have the correct Hankel structure is used as a means of increasing the relative signal strength. Also in signal processing, Hankel matrices with elements which are power sum occur in many applications. One of these applications involves the so-called frequencies of sinusoids problem, (see [32] and references therein). In general, Hankel matrices appear naturally in a variety of problems of engineering interest: communication, control engineering, filter design, identification, model reduction and broadband matching. Besides these regions of engineering applications, there is still one more section of mathematics in which Hankel matrices play the role of distinctive models. The point is, that the continual analogous of systems of linear algebraic equations, in which the

matrices of the coefficients are Hankel matrices, are integral equations with kernels, which depend on the sum of the arguments, (see [25], Section 18). Related problems occur in many engineering and statistics applications [11].

This problem was studied by Macinnes [32]. He proposed a method for finding the best approximation of a matrix A by a full rank Hankel matrix. He used Grassman coordinates to transform the initial problem to a problem involving best approximation of a given vector by a second vector whose elements are constrained so that its inverse image is a Hankel matrix. A similar problem with Hankel matrix replaced by Toeplitz-plus-Hankel matrix was solved by Fang et. al. [16]. They used two versions of the projection method to achieve their goal. Grigoriadis et. al. employed alternating convex projection techniques to compute the closest positive definite Toeplitz matrix that satisfies certain inequality constraints to a specified symmetric matrix. They discussed applications to signal processing and control problems. Other approximation problems were studied in [43, 40]. The problem was formulated as a nonlinear minimization problem with positive semidefinite Hankel matrix as constraints in [1] and then was solved by l_2Sequential Quadratic Programming (l_2SQP) method. Another approach to deal with this problem was to solve it as a smooth unconstrained minimization problem [2].

Our work is mainly casting the problem: first as a semidefinite programming problem and second as a mixed semidefinite and second-order cone optimization problem. A semidefinite programming (SDP) problem is to minimize a linear objective function subject to constraints over the cone of positive semidefinite

matrices. It is a relatively new field of mathematical programming, and most of the papers on SDP were written in 1990s, although its roots can be traced back to a few decades earlier (see Bellman and Fan [9]). SDP problems are of great interest due to many reasons, including:

- SDP contains important classes of problems as special cases, such as linear and quadratic programming.
- Important applications of SDP exist in combinatorial optimization, approximation theory, system and control theory, and mechanical and electrical engineering.
- Loosely speaking, SDP problems can be solved to ϵ-optimality in polynomial time by *interior point algorithms* [49, 52, 13, 7, 36].
- The availability of effective code and packages to solve large sized problems.
- SDP is a powerful modeling tool.

These features and other theoretical properties attracted our attention to this field. Fortunately, we managed to formulate our problem in three different ways each of which is an SDP with different problem sizes and different properties. The first uses the direct method of defining the distance between a given point and the desired one (SDV, see Section 3.3.1), the second employs the quadratic structure of the Frobenius norm (SDQ, see Section 3.3.3), and the third, which has the best impact of the three formulations, was developed using a new isometry operator, **hvec**, which reduces the SDP problem size while maintaining the properties of the initial problem by fully exploiting the Hankel structure our problem has (SDH, see Section 3.3.2).

The constraints in a mixed semidefinite and second-order cone optimization problem are constraints over the positive semidefinite and the second-order cones. Although the second-order cone constraints can be seen as positive semidefinite constraints, recent research has shown that it is more efficient to deal with mixed problems rather than the semidefinite programming problem. Nesterov et. al. [36] can be considered as the first paper to deal with mixed semidefinite and second-order cone optimization problems. However, the area was really brought to life by Alizadeh et. al. [6] with the introduction of SDPPack, a software package for solving optimization problems from this class. The practical importance of second-order programming was demonstrated by Lobo et. al. [30] and many subsequent papers. In [41] Sturm presented implementational issues of interior point methods for mixed SDP and SOCP problems in a unified framework. The recent trend to treat problems, if possible, as a mixture of semidefinite and second-order cone constraints rather than just semidefinite constraints leads us to think of the possibility of formulating our problem as a mixture one. Thus, we cast our problem as a mixed problem in three different ways; namely: SQV, SQQ and SQH. All of them have the same semidefinite part but different second-order cone part in terms of dimensions and definition. Although, as we said earlier, the first impression inherited from previous research is that mixed problems are more efficient than SDP, we noticed that SDH formulation perform better or at least as good as the mixed formulations. We think that this is due to the use of the economical vectorization operator, **hvec**.

One class of these interior point methods is the primal-dual path-following

methods. These methods are considered the most successful interior point algorithms for linear programming. Their extension from linear to semidefinite and then mixed problems has followed the same trends. Interior point methods enjoy several properties that make them attractive.

- **Practical efficiency**. It is now generally accepted that interior point methods for linear programs are competitive with the simplex method and even faster for problems with more than $10,000$ variables or constraints. As a very rough rule-of-thumb, interior point methods solve semidefinite programs in about 5-50 iterations; each iteration is basically a least-squares problem of the same size as the original problem.

- **Theoretical efficiency**. A worst-case analysis of interior-point methods for semidefinite programming shows that the effort required to solve a semidefinite program to a given accuracy grows no faster than a polynomial of the problem size.

- **Ability to exploit problem structure**. Most of the computational effort in an interior point method for semidefinite programming is in the least-squares problems that must be solved at each iteration. These least-squares problems can be solved by iterative methods such as conjugate-gradients, which can take advantage of problem structure. Sparsity is one well-known example of structure; in engineering applications many other types arise (e.g., Hankel and Toeplitz structure).

One of the most successful implementation of primal-dual path-following methods is in the software SDPT3 by Toh et al. [47, 44].

Similar problems, such as the problem of minimizing the spectral norm of a matrix was first formulated as a semidefinite programming problem in [49, 45]. Then, these problems and some others were formulated as a mixed semidefinite and second-order cone optimization problems [30, 5, 42].

1.2 Thesis Outline

We divided the thesis into six chapters

1. Introduction,
2. Projection Methods,
3. Semidefinite Programming,
4. Mixed Semidefinite and Second-order Cone Optimization (Mixed programming),
5. Interior Point Methods,
6. Numerical Results, and
7. Conclusions and Further Research.

For Chapters 2, 3, 4, and 5 we give a historical review for each one of the topics for which the Chapter is devoted. We did this instead of introducing a literature review in the introductory chapter for two reasons; One is to keep each chapter self-contained as much as possible. Second is the independence nature of each topic they involve, however, the range of intersection between some Chapters is quite wide especially Chapters 3 and 5, 4 and 5 which makes some repetition in

the historical aspects occur.

After introducing the problem of interest along with a brief literature survey and applications, the remaining of this introductory chapter will introduce the notations we adopt throughout our work as well as some preliminary concepts and results. Chapter 2 is devoted to the method of projection which is considered the most accurate method with a global convergent property. This chapter includes a description of von Neumann's and Dykstra's projection methods for solving least distance convex problems. We conclude this chapter by describing Dykstra's method when applied to our problem. This method will provide us with accurate results against which we can compare numerical results we obtain by solving the problem with the primal-dual path-following interior point method.

Semidefinite programming and mixed semidefinite and second-order cone optimization will be introduced in Chapters 3 and 4. A historical review will be given and some applications will be discussed. The primal and the dual of the respective optimization problem will be displayed. In each chapter, formulations of the problems in the form of respective optimization problem will be introduced .

Chapter 5 is concerned with interior point methods used to solve semidefinite programming and mixed problems. Indeed, we give a literature survey for the topic. Then we describe how a primal-dual path-following algorithm is derived for an SDP problem. We also give a generic scheme of such an algorithm and an example to illustrate the method.

We report some numerical results to practically see the performance of the primal-dual path-following algorithm on each formulations we proposed in Chapters 3 and 4. Then compare them, in terms of accuracy, with those we obtain from applying the projection method. Finally we state some of the conclusions of this work and we make some suggestions for further research in Chapter 7.

1.3 Notation

Throughout this thesis, Matrices are denoted by capital italic letters and their elements by small italic letters. Vectors and vectorization operators are denoted by small bold face letters. When we say operators we mean linear transformations. We denote the space of $m \times n$ real matrices by $\mathbb{R}^{m \times n}$. An $n \times n$ matrix A is symmetric if $A^T = A$ and is positive semidefinite (respectively, positive definite) if $\mathbf{x}^T A \mathbf{x} \geq 0, \forall \mathbf{x} \in \mathbb{R}^n$ (respectively, $\mathbf{x}^T A \mathbf{x} > 0, \forall \mathbf{x} \neq 0 \in \mathbb{R}^n$). The second-order cone of dimension k, $\mathcal{Q}_k$, is defined as

$$\mathcal{Q}_k = \{\mathbf{x} \in \mathbb{R}^k : \|\mathbf{x}_{2:k}\|_2 \leq x_1\},$$

(also called Lorentz cone, ice cream cone or quadratic cone).

The set of all $n \times n$ real Hankel matrices will be denoted by $\mathcal{H}_n$. An $n \times n$ real Hankel matrix H has the following structure:

$$H = \begin{bmatrix} h_1 & h_2 & \cdots & h_n \\ h_2 & h_3 & \cdots & h_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ h_n & h_{n+1} & \cdots & h_{2n-1} \end{bmatrix}.$$

It is clear that every Hankel matrix is symmetric. Other notations are listed below:

Matrix notation

$$
\begin{aligned}
A^T &: \text{transpose of } A \in \mathbb{R}^{m\times n},\\
A^{-T} &: (A^T)^{-1},\\
A_{ij} \text{ or } a_{ij} &: \text{the } ij^{\text{th}} \text{ entry of } A \in \mathbb{R}^{m\times n},\\
A \succeq B &: A-B \text{ is symmetric positive semidefinite},\\
A \succ B &: A-B \text{ is symmetric positive definite},\\
\mathbf{a} \geq_Q \mathbf{b} &: \mathbf{a}-\mathbf{b} \in \mathcal{Q}_k,\ \mathbf{a} \text{ and } \mathbf{b} \in \mathbb{R}^k,\\
\mathbf{a} \geq \mathbf{b} &: \text{each component of } \mathbf{a}-\mathbf{b} \text{ is nonnegative},\\
\mathbf{a} > \mathbf{b} &: \text{each component of } \mathbf{a}-\mathbf{b} \text{ is positive}.
\end{aligned}
$$

Special matrices

$$
\begin{aligned}
I &: \text{identity matrix of size depending on the context},\\
0 &: \text{zero vector/matrix of size depending on the context}.
\end{aligned}
$$

Sets of matrices and vectors

$$
\begin{aligned}
\mathbb{R}^n &: n\text{-dimensional real Euclidean vector space},\\
\mathcal{S}_n &= \{X : X \in \mathbb{R}^{n\times n}, X = X^T\},\\
\mathcal{S}_n^{+} &= \{X : X \in \mathcal{S}_n, X \succeq 0\},\\
\mathcal{S}_n^{++} &= \{X : X \in \mathcal{S}_n, X \succ 0\}.
\end{aligned}
$$

Functions of matrices and vectors

$$
\begin{aligned}
\mathbf{vec}(A) &= [a_{11}, a_{21}, \cdots, a_{12}, a_{22}, \cdots, a_{nn}]^T \text{ for } A \in \mathbb{R}^{n\times n},\\
\mathbf{hvec}(A) &= [a_{11}, \sqrt{2}a_{12}, \sqrt{3}a_{13} \cdots, \sqrt{n}a_{1n}, \sqrt{n-1}a_{n2}, \sqrt{n-2}a_{n3}, \cdots, a_{nn}]^T\\
&\quad \text{for } A \in \mathcal{H}_n,\\
\lambda_i(A) &: i^{\text{th}} \text{ largest eigenvalue of } A, \text{ if } \lambda_j(A) \in \mathbb{R}\ \forall j,\\
\lambda_{\min}(A) &= \min_i \lambda_i(A), \text{ if } \lambda_i(A) \in \mathbb{R}\ \forall i,\\
trace(A) &= \textstyle\sum_i a_{ii} \text{ (trace of } A \in \mathbb{R}^{n\times n}),\\
A \bullet B &= trace(AB^T) \text{ (Inner product)}\\
\|\mathbf{a}\|_2 &= \sqrt{\mathbf{a}^T\mathbf{a}} = \sqrt{\textstyle\sum_{i=1}^n a_i^2} \text{ (Euclidean norm)},\\
\|A\|_F^2 &= A \bullet A = \textstyle\sum_i \sum_j a_{ij}^2 = \|\mathbf{vec}(A)\|_2^2 \text{ (Frobenius norm)},\\
\rho(A) &= \max_i |\lambda_i(A)| \text{ (Spectral radius of } A),\\
A^{\frac{1}{2}} &: \text{unique symmetric square root factor of } A \succeq 0,\\
Diag(\mathbf{x}) &: n \times n \text{ diagonal matrix with components of } \mathbf{x} \in \mathbb{R} \text{ on diagonal},\\
diag(X) &: n\text{-vector obtained by extracting the diagonal of } X \in \mathbb{R}^{n\times n}.
\end{aligned}
$$

The end of a theorem statement, proposition or lemma will be marked $\square$. The end of a proof will be signaled by $\blacksquare$. The end of a definition statement will be marked $\triangle$.

Thus our problem in mathematical notation can now be formulated as follows: Given a data matrix $F \in \mathbb{R}^{n\times n}$, find the nearest positive semidefinite Hankel matrix H to F such that $\|F - H\|_F$ is minimal. Thus, we have the following

optimization problem:

$$\begin{aligned} \text{minimize} \quad & \|F - H\|_F \\ \text{subject to} \quad & H \in \mathcal{H}_n, \\ & H \succeq 0. \end{aligned} \tag{1.3.1}$$

1.4 Preliminaries

In this section, we list some well-known properties of positive (semi)definite matrices. We also give some background material on convex analysis which will be useful in this thesis.

Characterizations of positive (semi)definiteness:

Theorem 1.4.1

Let $X \in \mathcal{S}_n$. The following statements are equivalent:

- $X \in \mathcal{S}_n^+$ *or* $X \succeq 0$,
- $\mathbf{x}^T X \mathbf{x} \geq 0 \quad \forall \mathbf{x} \in I\!R^N$,
- $\lambda_{min}(X) \geq 0$,
- *All principal minors of X are nonnegative,*
- $X = LL^T$ *for some* $L \in I\!R^{n \times n}$.

□

We change 'positive semidefinite' by 'positive definite' in the statement of the theorem by changing the respective nonnegativity requirements to positivity, and by requiring that the matrix L in the last item be nonsingular. As a consequence of the theorem a block diagonal matrix is positive (semi)definite if and only if

each of its diagonal blocks is positive (semi)definite.

The trace operator: The trace of an $n \times n$ matrix A is defined as

$$trace(A) = \sum_{i=1}^{n} a_{ii}.$$

The trace is clearly a linear operator and has the following properties.

Theorem 1.4.2

Let $A \in I\!R^{n\times n}$ and $B \in I\!R^{n\times n}$. Then the following holds:

1. $trace(A) = \sum_{i=1}^{n} \lambda_i(A)$,
2. $trace(A) = trace(A^T)$,
3. $trace(AB) = trace(BA)$,
4. $trace(AB^T) = \boldsymbol{vec}^T(A)\boldsymbol{vec}(B) = \sum_{i,j=1}^{n} a_{ij}b_{ij}$.

□

The last item shows that we can view the usual Euclidean inner product on $I\!R^{n^2}$ as an inner product on $I\!R^{n\times n}$:

$$\langle A, B\rangle := trace(AB^T) = trace(B^TA) = trace(A^TB) = trace(B^TA).$$

Convex analysis:

Definition 1.4.1 (Convex set)

A set $S \in I\!R^n$ is said to be convex if the line segment joining any two points of the set also belongs to the set. In other words, if $x_1, x_2 \in S$, then $\lambda x_1 + (1-\lambda)x_2 \in S$

for each $\lambda \in [0,1]$.

The point $\lambda x_1 + (1-\lambda)x_2$ *is a convex combination of the two points* x_1 *and* x_2.△

Definition 1.4.2 (Convex function)

A function $f : S \to \mathbb{R}$ *defined on a convex set* S *is called convex if for all* $x_1, x_2 \in S$ *and* $\lambda \in [0,1]$ *we have*

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2).$$

The function is called strictly convex if the last inequality is strict. △

Strictly convex functions are useful for proving 'uniqueness properties' due to the following result.

Theorem 1.4.3

If a strictly convex function has a minimizer over a convex set, then this minimizer is unique. □

Definition 1.4.3 (Convex cone)

The set $\mathcal{K} \subset \mathbb{R}^n$ *is a convex cone if it is a convex set and for all* $x \in \mathcal{K}$ *and* $\lambda > 0$, *we have* $\lambda x \in \mathcal{K}$. △

Examples of convex cones are:

- The nonnegative orthant of $\mathbb{R}^n$:$\mathbb{R}^n_+ := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x} \geq 0\}$,
- The symmetric positive semidefinite matrices: $\mathcal{S}^+_n$,
- The second-order cone: $\mathcal{Q}_k$.

The following lemma is useful in finding the search directions of the primal-dual path-following algorithm in Chapter 5.

Lemma 1.4.1 Let $A_i \in \mathcal{S}_n \quad (i = 1, \cdots, m)$ be linearly independent, and let $0 \neq Y \in \mathcal{S}_n^+, Z \in \mathcal{S}_n^{++}$. The matrix $M \in \mathcal{S}_m$ with entries

$$m_{ij} := trace(A_i Z A_j Y), \quad i, j = 1, \cdots, m$$

is positive definite. □

Chapter 2

Projection Methods

2.1 Preamble

The aim of this chapter is to provide a mathematical background for the *projection methods* which were successfully used to solve problem (1.3.1), [1], and similar problems, [19, 20, 16]. Indeed, the projection methods can solve any *least-distance problem* of the following form:

$$\begin{aligned} \min \quad & \|\mathbf{a}-\mathbf{x}\| \\ \text{s.t.} \quad & \mathbf{x} \in \bigcap_{i=1}^{m} C_i. \end{aligned} \tag{2.1.1}$$

where $\mathbf{a}$ is some fixed element of an inner product space and $\bigcap_{i=1}^{m} C_i$ is the intersection of a finite number of closed convex sets $C_1, C_2, \cdots, C_m$.

These methods give accurate solutions and are globally convergent. However, the rate of convergence is slow (linear or sometimes less). This fact about projection methods gives us a tool to find an accurate solution for Problem (1.3.1) against which we can compare, in terms of accuracy, the results we obtain by applying the approach of semidefinite programming, Chapter 3, and the approach of cone-linear programming, Chapter 4.

Neumann (1950) [37], and independently Wiener [50], proved that successive cyclic projections onto the C_i's in (2.1.1), where the C_i's are closed subspaces, converge in a Hilbert space setting to the projection onto the intersection. Dykstra (1983) [15] developed a modification of Neumann's algorithm to deal with the case when each C_i is a convex cone. It was shown later by Boyle and Dykstra [10] that Dykstra's algorithm converges correctly in an infinite-dimensional Hilbert space setting even when each C_i is a convex set. The feasible region of (1.3.1) is the intersection of a subspace and a convex cone, namely: the subspace of $n \times n$ Hankel matrices $\mathcal{H}_n$ and the convex cone of $n \times n$ positive semidefinite matrices. Furthermore, the objective function is a least distance function. This makes Dykstra's algorithm a good candidate to solve it.

Some mathematical definitions and results concerning the projection methods will be given in Section 2. Section 3 will be devoted to Neumann's algorithm and its modification due to Dykstra along with their fundamental convergence theorems. The question: "How can we employ Dykstra's algorithm to solve problem (1.3.1)?" will be answered in Section 4.

2.2 Mathematical Background

It is well known that if C is a closed convex set and $\mathbf{y} \in \mathsf{H}$, (where H denotes a Hilbert space), and $\mathbf{y} \notin C$, then there exits a unique $\mathbf{x} \in C$ such that

$$\|\mathbf{x}-\mathbf{y}\| \;=\; \inf_{\mathbf{z}\in C}\|\mathbf{z}-\mathbf{y}\|. \tag{2.2.2}$$

This nearest point $\mathbf{x}$ is completely characterized by the "minimum principle" condition:

$$<\mathbf{x}-\mathbf{y},\mathbf{x}-\mathbf{z}> \leq 0, \qquad \forall\ \mathbf{z}\in C \tag{2.2.3}$$

Definition 2.2.1

We call the unique element $\mathbf{x}\in C$ satisfying (2.2.2) the Projection of $\mathbf{y}$ onto C, and is denoted as $P_C(\mathbf{y})$. △

When C is a subspace of H, $P_C(\mathbf{y})$ is simply the orthogonal projection onto C and we have only equality sign in (2.2.3). A closed form for $P_C(\mathbf{x})$ for any $\mathbf{x}\in\mathsf{H}$ can be found using the following result if C is a finite-dimensional subspace of H.

Theorem 2.2.1

Let C be a finite-dimensional subspace of a Hilbert space H and let $\mathbf{y}\in C$ and $\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n\}$ be any orthonormal basis for C. Then

$$P_C(\mathbf{y}) = \sum_{k=1}^{n}\frac{<\mathbf{y},\mathbf{x}_k>}{\|\mathbf{x}_k\|^2}\mathbf{x}_k. \tag{2.2.4}$$

□

Proof: (see [24]). ■

2.3 Dykstra's algorithm

Neumann's projection algorithm, also called *alternating projection algorithm*, generates a sequence of alternating projections in order to find the projection onto the feasible region of problem (2.1.1) when each C_i of $\bigcap C_i$ is a subspace of a Hilbert space H. Neumann showed that if C_1 and C_2 are subspaces and P_{C_1} and P_{C_2} are the orthogonal projections onto C_1 and C_2, respectively. Then the projection of a given point $\mathbf{a} \in \mathsf{H}$, $P_{C_1 \cap C_2}(\mathbf{a})$, can be found through the following algorithm:

Algorithm 2.3.1 (Neumann's Algorithm)

Let $\mathbf{x}_1 = \mathbf{a}$

For $k = 1, 2, \cdots$

$\mathbf{x}_{k+1} = P_{C_1}(P_{C_2}(\mathbf{x}_k))$.

Deutsch shows in [14] that the rate of convergence is linear and depends on the angle between the subspaces. In a natural way, Algorithm 2.3.1 can be generalized for m subspaces. Unfortunately, von Neumann's algorithm does not work generally, as the following example in $\mathbb{R}^2$ shows.

Example 2.3.1 (adapted from Han [21])

Let $C_1 = \{(x, y) : y \leq 0\}$, and $C_2 = \{(x, y) : x + y \leq 0\}$.

Define

$$P_{C_1}(x, y) = \begin{cases} (x, 0) & \text{if} \quad y > 0, \\ (x, y) & \text{otherwise.} \end{cases}$$

and

$$P_{C_2}(x, y) = \begin{cases} \frac{1}{2}(x - y, y - x) & \text{if} \quad x + y > 0, \\ (x, y) & \text{otherwise.} \end{cases}$$

It can be shown using the minimum condition principle that P_{C_1} and P_{C_2} are the orthogonal projections onto C_1 and C_2, respectively. Now, if $\mathbf{a} = (4, 3)$, then according to Algorithm 2.3.1, we have $\mathbf{x}_1 = P_{C_1}(a) = P_{C_1}(4, 3) = (4, 0)$ and then $\mathbf{x}_2 = P_{C_2}(4, 0) = (2, -2)$. At this point we stop. But this point is not the nearest point to $\mathbf{a}$, since $(0.5, -0.5)$ is closer.

Dykstra's algorithm is based on a simple modification of Neumann's algorithm. Given a collection of closed convex sets C_i, $i = 1, \cdots, m$, and the corresponding projections $P_i(\cdot)$ onto C_i and a point $\mathbf{x} \notin \bigcap_{i=1}^{m} C_i$, Dykstra's method constructs a sequence of vectors by the following algorithm.

Algorithm 2.3.2 (Dykstra's algorithm)

Set $\mathbf{y}_1^0, \mathbf{y}_2^0, \cdots, \mathbf{y}_m^0 = 0$

$\mathbf{x}_m^0$=**a**

$k = 1, 2, \cdots$

$\mathbf{x}_0^k = \mathbf{x}_m^{k-1}$

for $i = 1, \cdots, m$

$\mathbf{z}_i^{k-1} = \mathbf{x}_{i-1}^k + \mathbf{y}_i^{k-1}$

$\mathbf{x}_i^k = P_i(\mathbf{z}_i^{k-1})$

$\mathbf{y}_i^k = \mathbf{z}_i^{k-1} - P_i(\mathbf{z}_i^{k-1})$

This algorithm differs from Neumann's method in that prior to a projection onto C_i, the outward normal vector from the previous projection onto C_i is added to the current point. It can be shown that if one or more of the convex sets C_i are subspaces, then the addition of the normal vector is unnecessary for the corresponding projection. Thus, in case of all C_i being subspaces we recover Neumann's method.

Several convergence results have been proven concerning this algorithm. Dykstra [15], shows that if the C_i's are closed convex cones, then the $\mathbf{x}_i$ converges to the nearest point to $\mathbf{a}$ in the intersection of the C_i's. This will be sufficient for our purposes, but it should be noted that Boyle and Dykstra [10] have proven that Algorithm 2.3.2 works even for closed convex sets.

Figure 2.1 demonstrates how Algorithm 2.3.2 converges to the optimal solution

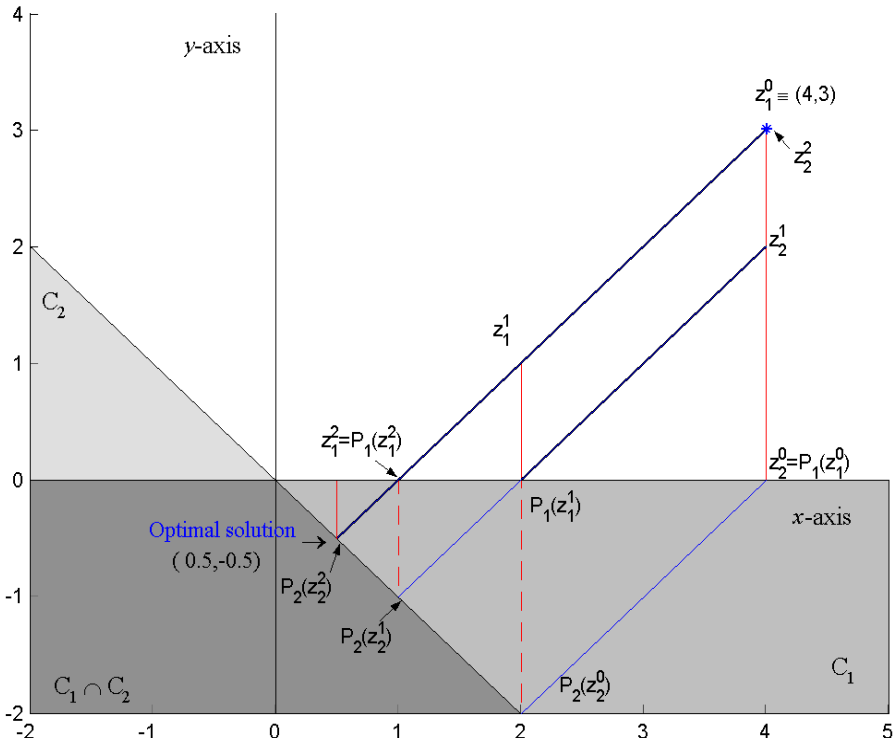

Figure 2.1: Demonstrating Algorithm 2.3.2

of Example 2.3.1 which is $(0.5, -0.5)$. Note that Neumann's algorithm stops after finding $P_1(z_1^0)$ and $P_2(z_2^0)$. But Dykstra's continues by computing the outer normal vector, $(0, 3)$, from the previous projection, *i. e.* z_2^0, and then adding it to the current point, $(2, -2)$. The resulting point is then projected onto C_1 to find $P_1(z_1^1)$. Again before projecting onto C_2 the outer normal vector, $(2, 2)$,

from $P_2(z_2^0)$ is added to $P_1(z_1^1)$ to obtain z_2^1 then projected onto C_2 to get $P_2(z_2^1)$. This process is repeated until convergence to the point $(0.5, -0.5)$ is established.

2.4 The nearest positive semidefinite Hankel matrix

In this section, we show how we can use Dykstra's algorithm 2.3.2 to solve (1.3.1). To do so, we need to put problem (1.3.1) in the form of problem (2.1.1). Note that the set of all real $n \times n$ symmetric positive semidefinite matrices denoted by $\mathcal{S}_n^+$ is a convex cone of dimension $n(n+1)/2$ and the set of all real $n \times n$ Hankel matrices denoted by $\mathcal{H}_n$ is a $2n-1$-dimensional subspace of the Hilbert space of all $n \times n$ real matrices equipped with the Frobenius matrix norm. Now, we can express (1.3.1) as

$$\begin{aligned} \min \quad & \|F - H\|_F \\ \text{st} \quad & H \in \mathcal{S}_n^+ \cap \mathcal{H}_n \end{aligned} \tag{2.4.5}$$

In addition, we need formulae for the projections onto $\mathcal{S}_n^+$ and $\mathcal{H}_n$. Let $F \in \mathbb{R}^{n\times n}$ be given. Then the projection onto $\mathcal{S}_n^+$, denoted $P_{\mathcal{S}_n^+}(\cdot)$, is given by

$$P_{\mathcal{S}_n^+}(F) = U\Lambda^+U^T, \tag{2.4.6}$$

where

$$\Lambda^+ = \begin{bmatrix} \Lambda_s & 0 \\ 0 & 0 \end{bmatrix},$$

and $\Lambda_s = \text{diag}[\lambda_1, \lambda_2, \cdots, \lambda_s]$ is the diagonal matrix formed from the positive eigenvalues of F. It is simply finding the spectral decomposition of F and setting the negative eigenvalues to zero.

In order to find the projection onto $\mathcal{H}_n$, we note that the $2n-1$ matrices $E_k,\ k = 1, \cdots, 2n-1$ where (i,j)-entry of each E_k is given by

$$E_k(i,j) = \begin{cases} 1 & \text{if } i+j = k+1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4.7}$$

form an orthonormal basis for $\mathcal{H}_n$. Hence, by Theorem 2.2.1, the orthogonal projection onto $\mathcal{H}_n$, denoted by $P_{\mathcal{H}_n}(\cdot)$, is given by

$$P_{\mathcal{H}_n}(F) = \sum_{k=1}^{2n-1} \frac{F \bullet E_k}{\|E_k\|_F^2} E_k. \tag{2.4.8}$$

One can compute $P_{\mathcal{H}_n}(F)$ simply by setting each antidiagonal of $P_{\mathcal{H}_n}(F)$ to the average of the corresponding antidiagonal of F.

To this end, we are ready to apply Dykstra's algorithm 2.3.2. However, we will propose an equivalent algorithm which is easier to program and cheaper to run taking advantage of $\mathcal{H}_n$ being a subspace. It is as follows: Given an arbitrary matrix $F \in \mathbb{R}^{n\times n}$, we have

Algorithm 2.4.1

Let $F^{(0)} = P_{\mathcal{H}_n}(F)$

For $k = 1, 2, \cdots$

$$F^{(k+1)} = F^{(k)} + P_{\mathcal{H}_n}(P_{\mathcal{S}_n^+}(F^{(k)})) - P_{\mathcal{S}_n^+}(F^{(k)}).$$

The equivalence between Algorithm 2.3.2 and Algorithm 2.4.1 can be easily shown by mathematical induction. The convergence theorem of Algorithm 2.3.2 due to Boyle and Dykstra [10] implies that the sequences $P_{\mathcal{H}_n}(P_{\mathcal{S}_n^+}(F^{(k)}))$ and $P_{\mathcal{S}_n^+}(F^{(k)})$ converge to the optimal solution of (2.4.5) and hence to the optimal solution of (1.3.1).

One final remark before we conclude this chapter relating to the computation of the projection $P_{\mathcal{S}_n^+}(F)$. Finding the spectral decomposition of F which is required to compute this projection is $\mathcal{O}(n^3)$ work (*i. e.* It requires up to $\mathcal{O}(n^3)$ floating-point operations per iteration) and hence $P_{\mathcal{S}_n^+}(F)$ is the major computational work of the algorithm. Another idea, should be investigated, is to find this projection by estimating the eigenvalues of F rather than finding them exactly. A method from numerical linear algebra can do that. It is called "Lanczos iteration". This method is only $\mathcal{O}(n^2)$ work.

Chapter 3

Semidefinite Programming

3.1 Preamble

In *semidefinite programming* (SDP) one minimizes a linear function subject to the constraint that an affine combination of symmetric matrices is positive semidefinite. Such a constraint is nonlinear and nonsmooth, but convex, so semidefinite programs are convex optimization problems. Semidefinite programming unifies several standard problems (e.g., linear and quadratic programming) and finds many applications in engineering and combinatorial optimization.

Although semidefinite programs are much more general than linear programs, they are not much harder to solve. Most interior point methods for linear programming have been generalized to semidefinite programs. As in linear programming, these methods have polynomial worst-case complexity, and perform very well in practice.

Semidefinite programming is a relatively new field of mathematical programming, and most of the papers on SDP were written in 1990s, although its roots can be traced back to a few decades further (see Bellman and Fan [9]). Other references discussing optimality conditions are Craven and Mond [12], Shapiro

[39], Fletcher [17], Allwright [8], Wolkowicz [51], and Kojima, Kojima and Hara [28].

Our coverage will necessarily be incomplete. Let us therefore refer the reader to a survey paper by Vandenberghe and Boyd [49] which discusses in particular a number of applications, especially in control theory. Another paper giving a number of examples demonstrating the significance of SDP and outlining duality theory is due to Todd [45]. The *Handbook of Semidefinite Programming* [52] contains an extensive review of both the theory and applications of SDP up to the year 2000. Most recently, a self-contained, to a large degree, book [13] treats SDP in a simple manner to an extent where it can be used for a graduate course on SDP. The area is receiving so much attention so that it is hard to keep abreast of recent development , but this is immeasurably assisted by two web sites that of Helmberg [22] and that of Alizadeh [3].

In this chapter we will see that Problem (1.3.1) can be cast as an SDP problem. This was motivated by the development of efficient algorithms, which solve SDP problems very efficiently both in theory and practice. Another key motivation was the power of SDP to model problems arising in a very wide range of areas.

The remainder of this chapter will be organized as follows: In Section 2 we describe the standard primal and dual SDP problems giving some examples showing the importance of SDP. In Section 3 we put problem (1.3.1) in the form of an SDP problem.

3.2 SDP problems

The semidefinite programming (SDP) problem in *primal standard form* is:

$$\begin{aligned} (P) \quad & \min_X C \bullet X \\ & \text{s. t. } A_i \bullet X = b_i, \quad i = 1, \cdots, m \\ & X \succeq 0, \end{aligned} \tag{3.2.1}$$

where all $A_i, C \in \mathcal{S}_n, b \in I\!R^m$ are given, and $X \in \mathcal{S}_n$ is the variable. This optimization problem is a convex optimization problem since its objective and constraints are convex.

We also consider SDP in *dual standard form*:

$$\begin{aligned} (D) \quad & \max_{\mathbf{y}} \mathbf{b}^T \mathbf{y} \\ & \text{s. t.} \\ & \sum_{i=1}^m y_i A_i + S = C \\ & S \succeq 0, \end{aligned} \tag{3.2.2}$$

where $\mathbf{y} \in I\!R^m$ and $S \in \mathcal{S}_n$ are the variables. This can be written as

$$\begin{aligned} & \max_{\mathbf{y}} \mathbf{b}^T \mathbf{y} \\ & \text{s. t.} \\ & \sum_{i=1}^m y_i A_i \preceq C, \end{aligned} \tag{3.2.3}$$

The first dual form with the "slack matrix" S will be useful especially when we discuss algorithms in Chapter 5. The second dual form (3.2.3) will be used throughout this chapter due to its simplicity and also the flexibility it provides

for modeling.

Although the SDP problem (3.2.3) may appear quite specialized, it includes many important optimization problems as special cases. For instance, consider the linear program (LP)

$$\begin{aligned} \min \ & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \ & A\mathbf{x} + \mathbf{b} \geq 0 \end{aligned} \tag{3.2.4}$$

in which the inequality denotes component wise inequality. Since a vector $\mathbf{v} \geq 0$ if and only if $\text{diag}(\mathbf{v}) \succeq 0$ (i.e., the diagonal matrix with the components of $\mathbf{v}$ on its diagonal) is positive semidefinite, we can express the LP (3.2.4) as a dual SDP problem (3.2.3) with

$$\mathbf{b} = \mathbf{c},\ C = \text{diag}(\mathbf{b}),\ A_i = -\text{diag}(a_i),\ \ i = 1, \cdots, m;$$

where $A = [a_1, \cdots, a_m] \ \in I\!R^{n\times m}$.

To introduce other examples, we have to present the following useful theorem.

Theorem 3.2.1 (Schur Complement)

If

$$M = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$$

where $A \in \mathcal{S}_n^{++}$ and $C \in \mathcal{S}_n$, then the matrix M is positive (semi)definite if and only if the matrix $C - B^TA^{-1}B$ is positive (semi)definite. □

The matrix $C - B^TA^{-1}B$ is called the Schur complement of A in M.

Proof:

The result follows by setting $D = -A^{-1}B$, and noting that

$$\begin{bmatrix} I & 0 \\ D^T & I \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} I & D \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & C - B^T A^{-1} B \end{bmatrix}.$$

Since a block diagonal matrix is positive (semi)definite if and only if its blocks are positive (semi)definite, the proof is complete. ■

Now, we introduce the so-called general convex quadratically constrained quadratic program (QCQP)

$$\begin{aligned} \min \ & f_0(\mathbf{x}) \\ \text{s.t.} \ & f_i(\mathbf{x}) \leq 0, \quad i = 1, \cdots, L, \end{aligned} \tag{3.2.5}$$

where each f_i is a convex quadratic function and $f_i(\mathbf{x}) = \mathbf{x}^T B_i \mathbf{x} - 2\mathbf{c}_i^T \mathbf{x} - d_i$, $B_i \in \mathcal{S}_n^+$. Assume for simplicity that $B_i \in \mathcal{S}_n^{++}$, hence, let $B_i = B_i^{1/2} B_i^{1/2}$. Then using Theorem 3.2.1, Problem (3.2.5) can be written as

$$\begin{aligned} \min \ & t \\ \text{s.t.} \ & \begin{bmatrix} I & B_i^{1/2}\mathbf{x} \\ (B_i^{1/2}\mathbf{x})^T & \mathbf{c}_i^T\mathbf{x} + d_i + t \end{bmatrix} \succeq 0, \\ & \begin{bmatrix} I & B_i^{1/2}\mathbf{x} \\ (B_i^{1/2}\mathbf{x})^T & \mathbf{c}_i^T\mathbf{x} + d_i \end{bmatrix} \succeq 0. \end{aligned}$$

which is an SDP problem in the dual form with $\mathbf{x} \in I\!R^n$ and $t \in I\!R$ as variables. This SDP problem has dimensions $n + 1$ and $L \times n + L + 1$.

3.3 SDP Formulations

Inspired by the above examples, we will formulate Problem (1.3.1) as an SDP problem in the dual form (3.2.3). To do so, we need to use some tools. The following theorem, which can be considered as a corollary of Theorem (3.2.1), provide these tools.

Theorem 3.3.1

Let $\mathbf{a}(x) \in I\!R^n$ *depend affinely on* x*. Then the following minimization problem:*

$$min \quad \|\mathbf{a}(x)\|_2,$$

cab be solved by solving the following SDP problem:

$$min \;\; t, \;\; s.t. \;\; \begin{bmatrix} I & \mathbf{a}(x) \\ (\mathbf{a}(x))^T & t \end{bmatrix} \succeq 0,$$

where t *is a nonnegative real scalar.* □

Proof:

Since $\|\mathbf{a}\|_2 = \sqrt{\mathbf{a}^T\mathbf{a}}$, we may minimize $\|\mathbf{a}\|_2$ by minimizing $\mathbf{a}^T\mathbf{a}$. So, let $\mathbf{a}^T\mathbf{a} \leq t$. Hence, $t - \mathbf{a}^T I\mathbf{a} \succeq 0$. So, by Theorem (3.2.1) the proof is complete. ■

3.3.1 Formulation I (SDV)

We are now ready to introduce the first formulation of (1.3.1) as an SDP problem.

We have

$$\|F - H\|_F = \|\mathbf{vec}(F - H)\|_2$$

So by Theorem 3.3.1, Problem (1.3.1) is cast as

$$\begin{aligned} (SDV) \quad & \min \ t \\ & \text{s.t.} \\ & \begin{bmatrix} t & 0 & 0 \\ 0 & H & 0 \\ 0 & 0 & V \end{bmatrix} \succeq 0, \end{aligned} \tag{3.3.6}$$

where

$$V = \begin{bmatrix} I & \mathbf{vec}(F - H) \\ \mathbf{vec}^T(F - H) & t \end{bmatrix}$$

and $t \in \mathbb{R}^+$. Problem (3.3.6) is an SDP problem in dual form (3.2.3) with dimensions $2n$ and n^2+n+2. To see this, we identify

$$y_1 = t, \quad y_k = h_{k-1}, \quad k = 2, \cdots, 2n$$

$$\mathbf{b} = \begin{bmatrix} -1 & 0 & \cdots & 0 \end{bmatrix}^T,$$

$$A_1 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix},$$

$$A_k = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -E_{k-1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{vec}(E_{k-1}) \\ 0 & 0 & \mathbf{vec}^T(E_{k-1}) & 0 \end{bmatrix}, \; k = 2, \cdots, 2n$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & I & \mathbf{vec}(F) \\ 0 & 0 & \mathbf{vec}^T(F) & 0 \end{bmatrix},$$

(The matrices E_{k-1} are defind in (2.4.7)). SDP problem (3.3.6) is very large even for a small data matrix F. For example, a 50×50 matrix F will give rise to a problem with dimensions 100 and 2552, hence solving (1.3.1) using formulation (3.3.6) is not efficient. Furthermore, we do not exploit the structure of H being Hankel explicitly. The structure of H is embedded in the other constraints. This discussion leads us to think of another way of formulation that produces an SDP problem with reasonable dimensions and exploits the Hankel structure of

H. This can be done by means of the following isometry operator, which seems to be new:

Definition 3.3.1

Let $\boldsymbol{hvec} : \mathcal{H}_n \longrightarrow I\!R^{2n-1}$ *be defined as*

$\boldsymbol{hvec}(U) = [u_{11}, \sqrt{2}u_{12}, \sqrt{3}u_{13}, \cdots, \sqrt{n}u_{1n}, \sqrt{n-1}u_{n2}, \cdots, u_{n,n}]^T$ *for any* $U \in \mathcal{H}_n$. △

One can easily show that **hvec** is a linear operator from the set of all $n \times n$ real Hankel matrices to $I\!R^{2n-1}$. The following theorem gives us some characterizations of **hvec**.

Theorem 3.3.2

For the operator ***hvec*** *, defined in Definition 3.3.1, the following conditions hold: For any* $U, V \in \mathcal{H}_n$

1. $U \bullet U = \boldsymbol{hvec}^T(U)\boldsymbol{hvec}(U)$.
2. $\|U - V\|_F^2 = \boldsymbol{hvec}^T(U - V)\boldsymbol{hvec}(U - V)$. □

Proof:

Part 1 is clear from the definition of the **hvec** operator. Part 2 is a consequence of part 1. ■

Part 1 implies that **hvec** is an isometry. We can not take any advantage of the operator and the theorem unless F is Hankel. We can make F Hankel by projecting it onto $\mathcal{H}_n$ using the orthogonal projection, $P_{\mathcal{H}_n}$, in Section 2.4 to get a Hankel matrix, say $\hat{F}$. But, is the nearest Hankel positive semidefinite matrix to $\hat{F}$, the nearest to F? The following proposition gives the answer.

Proposition 3.3.1 Let $\hat{F}$ be the orthogonal projection of F onto $\mathcal{H}_n$ and let H be the nearest Hankel positive semidefinite matrix to $\hat{F}$, then H is so for F. □

Proof:

If $\hat{F}$ is positive semidefinite, then we are done. If not, then for any $T \in \mathcal{H}_n$, we have

$$(F - \hat{F})^T \bullet (\hat{F} - T) = 0$$

since $\hat{F}$ is the orthogonal projection of F. Thus,

$$\|F - T\|_F^2 = \|F - \hat{F}\|_F^2 + \|\hat{F} - T\|_F^2$$

and since we are minimizing $\|\hat{F} - T\|_F$, therefore again we are done. ■

As a consequence of this proposition, we have the following problem equivalent to (1.3.1):

$$\begin{aligned} \text{minimize} \quad & \|\hat{F} - H\|_F \\ \text{subject to} \quad & H \in \mathcal{H}_n, \\ & H \succeq 0. \end{aligned} \tag{3.3.7}$$

3.3.2 Formulation II (SDH)

From Theorem 3.3.1 and part 2 of Theorem 3.3.2, we have the following SDP problem for $t \geq 0, t \in I\!R$:

$$\begin{aligned} (SDH) \quad & \min \ t \\ & \text{s.t.} \\ & \begin{bmatrix} t & 0 & 0 \\ 0 & H & 0 \\ 0 & 0 & \hat{V} \end{bmatrix} \succeq 0 \end{aligned} \tag{3.3.8}$$

where

$$\hat{V} = \begin{bmatrix} I & \mathbf{hvec}(\hat{F} - H) \\ \mathbf{hvec}^T(\hat{F} - H) & t \end{bmatrix},$$

This SDP problem has dimensions $2n$ and $3n+1$ which is far better than (3.3.6).

3.3.3 Formulation III (SDQ)

Another way for formulating (1.3.1) is through the definition of the Frobenius norm being a quadratic function. Indeed,

$$\|F - H\|_F^2 = \mathbf{y}^T P \mathbf{y} + 2\mathbf{q}^T \mathbf{y} + r,$$

where

$$\mathbf{y} = [h_1\ h_2\ \cdots\ h_{2n-1}]^T,$$
$$P = \mathrm{diag}([1\ 2\ \cdots\ n\ \cdots 2\ 1]),$$
$$\mathbf{q}_k = -\sum_{\substack{i,j=1 \\ i+j=k+1}}^{n} F(i,j), \quad k = 1, 2, \cdots 2n-1 \ \text{ and}$$
$$r = \|F\|_F^2.$$

Now, we have for a nonnegative real scalar t,

$$\begin{aligned}
& \|F - H\|_F^2 \le t \\
\Leftrightarrow\ & \mathbf{y}^T P \mathbf{y} + 2q^T \mathbf{y} + r \le t \\
\Leftrightarrow\ & (P^{1/2}\mathbf{y})^T (P^{1/2}\mathbf{y}) + 2q^T \mathbf{y} + r \le t \\
\Leftrightarrow\ & t - 2q^T \mathbf{y} - r - (P^{1/2}\mathbf{y})^T I (P^{1/2}\mathbf{y}) \ge 0 \\
\Leftrightarrow\ & \begin{bmatrix} I & (P^{1/2}\mathbf{y}) \\ (P^{1/2}\mathbf{y})^T & t - 2q^T \mathbf{y} - r \end{bmatrix} \succeq 0.
\end{aligned}$$

Hence, we have the following SDP problem:

$$\begin{aligned}
(SDQ) \quad & \min\ t \\
& \text{s.t.} \\
& \begin{bmatrix} t & 0 & 0 \\ 0 & H & 0 \\ 0 & 0 & Q \end{bmatrix} \succeq 0,
\end{aligned} \tag{3.3.9}$$

where

$$Q = \begin{bmatrix} I & (P^{1/2}\mathbf{y}) \\ (P^{1/2}\mathbf{y})^T & t - 2q^T\mathbf{y} - r \end{bmatrix},$$

This SDP problem is of dimensions $2n$ and $3n+1$. Although problem (3.3.9) has the same dimensions as the problem (3.3.8), it is less efficient to solve it over the positive semidefinite cone $\mathcal{S}_n^+$ especially when we have large size F. In practice, as we will see in Chapter 6, it has been found that the performance of this formulation is poor; because the matrix P is of full rank. A more efficient interior point method for this formulation can be developed by using Nesterov and Nemirovsky formulation as a problem over the second-order cone (see [34] Section 6.2.3). This is what we will see in the next chapter.

To illustrate how we can use the formulations SDV, SDH and SDQ to model Problem (1.3.1), we consider the following example:

Example 3.3.1

Consider Problem (1.3.1) with

$$F = \begin{bmatrix} -4 & 2 & 1 \\ -6 & -1 & 0 \\ 3 & 2 & 7 \end{bmatrix}$$

and let $t \in \mathbb{R}^+$. Then we have

- SDV: The SDV formulation is

$$\begin{aligned}
&\min \ t \\
&\text{s.t.} \\
&\begin{bmatrix}
t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_2 & h_3 & h_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & s_1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & s_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & s_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & s_4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & s_5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & s_6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & s_7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & s_8 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & s_9 \\
0 & 0 & 0 & 0 & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 & s_8 & s_9 & t
\end{bmatrix} \succeq 0,
\end{aligned}$$

where

$$\mathbf{s} = [s_i]_{i=1}^{9} = \begin{bmatrix} -4 - h_1 \\ -6 - h_2 \\ 3 - h_3 \\ 2 - h_2 \\ -1 - h_3 \\ 2 - h_4 \\ 1 - h_3 \\ -h_4 \\ 7 - h_5 \end{bmatrix}$$

or equivalently,

$$\begin{aligned} &\max \quad -t \\ &\text{s.t.} \\ &A_1 t + A_2 h_1 + A_3 h_2 + A_4 h_3 + A_5 h_4 + A_6 h_5 \preceq C \end{aligned}$$

where

$$A_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

and for $k = 2, \cdots, 6$ we have

$$A_k = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -E_{k-1} & 0 & 0 \\ 0 & 0 & I & \mathbf{vec}(E_{k-1}) \\ 0 & 0 & \mathbf{vec}^T(E_{k-1}) & 0 \end{bmatrix}$$

here, E_k is the matrix defined in (2.4.7). Finally

$$C = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 7 \\
0 & 0 & 0 & 0 & -4 & -6 & 3 & 2 & -1 & 2 & 1 & 0 & 7 & 0
\end{bmatrix}$$

- SDH: Before we construct this formulation we find the orthogonal projection of F onto $\mathcal{H}_n$

$$\hat{F} = P_{\mathcal{H}_n}(F) = \begin{bmatrix} -4 & -2 & 1 \\ -2 & 1 & 1 \\ 1 & 1 & 7 \end{bmatrix}$$

Now, we find

$$\mathbf{hvec}(\hat{F}) = \begin{bmatrix} -4 \\ -2\sqrt{2} \\ \sqrt{3} \\ \sqrt{2} \\ 7 \end{bmatrix},$$

and

$$\mathbf{s} = [s_i]_{i=1}^5 = \mathbf{hvec}(\hat{F} - H) = \begin{bmatrix} -4 - h_1 \\ \sqrt{2}(-2 - h_2) \\ \sqrt{3}(1 - h_3) \\ \sqrt{2}(1 - h_4) \\ 7 - h_5 \end{bmatrix}.$$

Hence, the SDH formulation is

$$\begin{aligned} &\min \ t \\ &\text{s.t.} \\ &\begin{bmatrix} t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_2 & h_3 & h_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & s_1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & s_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & s_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & s_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & s_5 \\ 0 & 0 & 0 & 0 & s_1 & s_2 & s_3 & s_4 & s_5 & t \end{bmatrix} \succeq 0, \end{aligned}$$

This is in the form (3.2.2) with

$$b = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix},$$

and for $k = 2, \cdots, 6$ we have

$$A_k = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -E_{k-1} & 0 & 0 \\ 0 & 0 & I & \mathbf{hvec}(E_{k-1}) \\ 0 & 0 & \mathbf{hvec}^T(E_{k-1}) & 0 \end{bmatrix}.$$

The matrices E_k's are as before.

- SDQ: For this formulation we have

$$\|F - H\|_F^2 = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix}$$
$$+ 2 \begin{bmatrix} -4 & -2 & 1 & 1 & 7 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix} + 78.$$

The SDQ formulation is

$$\begin{aligned}
&\min \ t \\
&\text{s.t.} \\
&\begin{bmatrix}
t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_2 & h_3 & h_4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & h_1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \sqrt{2}h_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \sqrt{3}h_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \sqrt{2}h_4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & h_5 \\
0 & 0 & 0 & 0 & h_1 & \sqrt{2}h_2 & \sqrt{3}h_3 & \sqrt{2}h_4 & h_5 & \tau
\end{bmatrix} \succeq 0,
\end{aligned}$$

where $\tau = t - 8h_1 + 4h_2 - 2h_3 - 2h_4 - 7h_5 - 78$. This problem can be seen as an SDP problem in the dual form in the same manner we have seen the other two formulations.

The last formulation seems to be straightforward, however it was found that using this formulation to solve similar problems was not a good idea. The reasons for that will be discussed in the following chapter when we talk about second-order cone programming. This fact about SDQ formulation will be clear in Chapter 6 when we use it to solve numerical examples, especially for large size F. We think also SDV formulation is not good enough to compete with other formulations even with the projection method. This is simply due to the fact that the amount of work per one iteration of interior-point methods that solve SDV formulation is

$\mathcal{O}(n^6)$, where n in the dimension of F. This disappointing fact makes using SDV formulation to solve (1.3.1) a waste of time. This leaves us with SDH formulation from which we expect good performance; since it does not suffer from the illness of SDQ nor the huge size of SDV.

Chapter 4

Mixed Semidefinite and Second-order Cone Programming

4.1 Preamble

In a *second-order cone program* (SOCP) a linear function is minimized over the intersection of an affine set and the product of second-order (quadratic) cones. SOCPs are nonlinear convex problems that include linear and (convex) quadratic programs as special cases. On the other hand semidefinite programming (SDP), as we have seen in the previous chapter, includes SOCP as a special case. Therefore, SOCP falls between linear (LP) and quadratic (QP) programming and SDP programming. Like LP, QP and SDP problems, SOCP problems can be solved in polynomial time by interior point methods. The computational effort per iteration required by these methods to solve SOCP problem is greater than that required to solve LP and QP problems but less than that required to solve SDP's of similar size and structure.

While SOCP problems can be solved as SDP problems, doing so is not advisable both on numerical grounds and computational complexity concerns. Particular examples of SOCP problems have been studied for a long time. The paper of Lobo et. al. [30] contains many applications of SOCP in engineering. Nesterov and Nemirovski [34] showed that many kinds of problems can be formulated as SOCPs.

Recent research has shown that it is more efficient to deal with mixed SDP and SOCP problems rather than SDP problems only. Nesterov et. al. [36] can be considered as the first paper to deal with mixed semidefinite and second-order cone optimization problems. However, the area was really brought to life by Alizadeh et. al. [6] with the introduction of SDPPack, a software package for solving optimization problems from this class. The practical importance of second-order programming was demonstrated by Lobo et al. [30] and many subsequent papers. In [41], Sturm presented implementational issues of interior point methods for mixed SDP and SOCP problems in a unified framework.

We show in this chapter that Problem (1.3.1) is nothing but another example of mixed SDP and SOCP. As far as we know our work is the first attempt to solve (1.3.1) as a mixed SDP and SOCP.

In Section 2 of this chapter we introduce a standard form for SOCP problems and we show that linear programming (LP) and quadratically constrained quadratic programming (QCQP) can be cast as an SOCP and we address some remarks on the advantage of this over expressing them as an SDP. The primal

and the dual problems of a mixed SDP and SOCP are introduced in Section 3. Section 4 is devoted to the formulation of (1.3.1) as a mixed SDP and SOCP problem along with some concluding remarks.

4.2 Second-Order Cone Programming

We consider the *second-order cone program* (SOCP)

$$\begin{aligned} &\min \ \ f^T\mathbf{x} \\ &\text{s.t.} \ \ \|A_i\mathbf{x}+\mathbf{b}_i\| \le \mathbf{c}_i^T\mathbf{x}+d_i, \quad i=1,\cdots,N, \end{aligned} \tag{4.2.1}$$

where $\mathbf{x} \in I\!R^n$ is the optimization variable, and $f \in I\!R^n$, $A_i \in I\!R^{(n_i-1)\times n}$, $\mathbf{b}_i \in I\!R^{n_i-1}$, $\mathbf{c}_i \in I\!R^n$, and $d_i \in I\!R$ are given data. The norm appearing in the constraint is the standard Euclidean norm, *i. e.* , $\|\mathbf{u}\| = \left(\mathbf{u}^T\mathbf{u}\right)^{1/2}$. We call the constraint

$$\|A_i\mathbf{x}+\mathbf{b}_i\| \le \mathbf{c}_i^T\mathbf{x}+d_i,$$

a *second-order cone constraint of dimension* n_i, simply because

$$\|A_i\mathbf{x}+\mathbf{b}_i\| \le \mathbf{c}_i^T\mathbf{x}+d_i \iff \begin{bmatrix} \mathbf{c}_i^T \\ A_i \end{bmatrix}\mathbf{x} + \begin{bmatrix} d_i \\ \mathbf{b}_i \end{bmatrix} \in \mathcal{Q}_{n_i}$$

Recall that a second-order cone of dimension n_i is defined as

$$\mathcal{Q}_{n_i} = \left\{ \begin{bmatrix} t \\ \mathbf{u} \end{bmatrix} : \mathbf{u} \in I\!R^{n_i-1},\ t \in I\!R,\ \|\mathbf{u}\| \le t \right\},$$

and hence the set of points satisfying a second-order cone constraint is convex. Thus, the SOCP (4.2.1) is a convex programming problem since the objective is convex function and the constraints define a convex set.

Second-order cone constraints can be used to represent several common convex constraints. For example, when $n_i = 1$ for $i = 1, \cdots, N$, the SOCP (4.2.1) reduces to the LP problem:

$$\begin{aligned} &\min \ f^T \mathbf{x} \\ &\text{s.t. } 0 \leq \mathbf{c}_i^T \mathbf{x} + d_i, \quad i = 1, \cdots, N. \end{aligned}$$

Another interesting example is convex quadratically constrained quadratic program (QCQP) (3.2.5, page 28). In this example we have the problem

$$\begin{aligned} &\min \ \mathbf{x}^T B_0 \mathbf{x} - 2\mathbf{c}_0^T \mathbf{x} - d_0 \\ &\text{s.t. } \mathbf{x}^T B_i \mathbf{x} - 2\mathbf{c}_i^T \mathbf{x} - d_i \leq 0, \quad i = 1, \cdots, L. \end{aligned} \tag{4.2.2}$$

This problem can be rewritten as

$$\begin{aligned} &\min \ \|B_0^{1/2}\mathbf{x} - B_0^{-1/2}\mathbf{c}_0\|^2 - d_0 + \mathbf{c}_0^T B_0^{-1} \mathbf{c}_0 \\ &\text{s.t. } \|B_i^{1/2}\mathbf{x} - B_i^{-1/2}\mathbf{c}_i\|^2 - d_i + \mathbf{c}_i^T B_i^{-1} \mathbf{c}_i \leq 0, \quad i = 1, \cdots, L, \end{aligned}$$

which can be solved via the SOCP with $L + 1$ constraints of dimension $n + 1$

$$\begin{aligned} &\min \ t \\ &\text{s.t. } \|B_0^{1/2}\mathbf{x} - B_0^{-1/2}\mathbf{c}_0\| \leq t, \\ &\qquad \|B_i^{1/2}\mathbf{x} - B_i^{-1/2}\mathbf{c}_i\| \leq (d_i - \mathbf{c}_i^T B_i^{-1} \mathbf{c}_i)^{1/2}, \quad i = 1, \cdots, L, \end{aligned} \tag{4.2.3}$$

where $t \in I\!R$ is a new optimization variable. Problems (3.2.5) and (4.2.3) will have the same optimal solution and the same optimal values up to a constant.

This shows that SOCP contains interesting examples. On the other hand, it is itself contained in SDP. This can be seen by the following property which is true for each vector $\mathbf{u}$ and scalar t:

$$\|\mathbf{u}\| \leq t \iff \begin{bmatrix} tI & \mathbf{u} \\ \mathbf{u}^T & tI \end{bmatrix} \succeq 0,$$

using this property SOCP (4.2.1) can be expressed as SDP

$$\begin{aligned} \min \ & f^T\mathbf{x} \\ \text{s.t.} \ & \begin{bmatrix} (\mathbf{c}_i^T\mathbf{x} + d_i)I & A_i\mathbf{x} + b_i \\ (A_i\mathbf{x} + b_i)^T & \mathbf{c}_i^T\mathbf{x} + d_i \end{bmatrix} \succeq 0, \quad i = 1, \cdots, N. \end{aligned} \tag{4.2.4}$$

Solving SOCP via SDP is not a good idea, however. Interior point methods that solve the SOCP directly have a much better worst-case complexity than an SDP method applied to (4.2.1): the number of iterations to decrease the duality gap to a constant fraction of itself is bounded above by $\mathcal{O}(\sqrt{N})$ for the SOCP algorithm, and by $\mathcal{O}(\sqrt{\sum_i n_i})$ for the SDP algorithm (see [34]). More importantly in practice, each iteration is much faster: the amount of work per iteration is $\mathcal{O}(n^2 \sum_i n_i)$ in the SOCP algorithm and $\mathcal{O}(n^2 \sum_i n_i^2)$ for the SDP. The difference between these numbers is significant if the dimensions n_i of the second-order constraints are large.

Returning to Problem (4.2.1) we see that it may be written in such a way to have a constraint over the positive semidefinite cone and a constraint over the second-order cone, namely: $H \succeq 0$ and $\|F - H\|_F \leq t$, respectively. Thus, in

order to take advantage of the good behavior of the SOCP we need to deal with mixed SDP and SOCP problems. Fortunately, interior point methods that solve such mixed problems are available. Indeed, these methods are the same as those of SDP with slight modifications. We will discuss these methods in the coming chapter. But now let us study our problem in a more general framework.

4.3 Cone-Linear Programming

A *cone-linear programming* (Cone-LP) is a unified way to study SDP and SOCP problems. The standard canonical form of Cone-LP problems is

$$\min \ \mathbf{c}^T\mathbf{x} \ \text{ s.t. } \ A\mathbf{x} = \mathbf{b}, \ \mathbf{x} \in \mathcal{K}, \tag{4.3.5}$$

where $\mathbf{x} \in I\!R^n$ is the vector of decision variables, $\mathcal{K} \subset I\!R^n$ is a pre-specified convex cone, and $\mathbf{b} \in I\!R^m$, $\mathbf{c} \in I\!R^n$ and $A \in I\!R^{m\times n}$ are given data. Despite its name, Cone-LP is non-linear, since $\mathcal{K}$ need not be polyhedral.

Important subclasses of Cone-LP are linear programming, semidefinite programming, second-order cone programming, and a mixture of these. These subclasses arise by letting $\mathcal{K}$ in (4.3.5) be the nonnegative orthant $\mathcal{K} = I\!R^n_+$, the cone of positive semidefinite matrices $\mathcal{S}^+_n$, the second-order cone $\mathcal{Q}_n$, or a mixture of them, respectively.

A mixed semidefinite and second-order cone optimization problem can be

formulated as a standard Cone-LP problem (4.3.5) with the following structure:

$$\begin{aligned} \min\ & C_S \bullet X_S + C_Q^T X_Q + C_L^T X_L \\ \text{s.t.}\ & (A_S)_i \bullet X_S + (A_Q)_i^T X_Q + (A_L)_i^T X_L = b_i, \quad i = 1, \cdots, m \\ & X_S \succeq 0, X_S \geq_Q 0, X_L \geq 0, \end{aligned} \tag{4.3.6}$$

where $X_S \in \mathcal{S}_n$, $X_Q \in I\!R^k$ and $X_L \in I\!R^{n_L}$ are the variables. C_S, $(A_S)_i \in \mathcal{S}_n$, $\forall i$, C_Q, $(A_Q)_i \in I\!R^k$ $\forall i$ and C_L, $(A_L)_i \in I\!R^{n_L}$ $\forall i$ are given data. Each of the three inequalities has a different meaning: $X_S \succeq 0$ means, as we have seen, that $X_S \in \mathcal{S}_n^+$, $X_S \geq_Q 0$ means that $X_Q \in \mathcal{Q}_k$ and $X_L \geq 0$ means that each component of X_L is nonnegative. It is possible that one or more of the three parts of (4.3.6) is not present. If the second-order part is not present, then (4.3.6) reduces to the ordinary SDP (3.2.1) and if the semidefinite part is not present, then (4.3.6) reduces to the so-called convex quadratically constrained linear programming problem.

The standard dual of (4.3.6) is:

$$\begin{aligned} \max\ & \mathbf{b}^T \mathbf{y} \\ \text{s.t.}\ & \\ & \sum_{i=1}^m y_i (A_S)_i \preceq C_S \\ & \sum_{i=1}^m y_i (A_Q)_i \leq_Q C_Q \\ & \sum_{i=1}^m y_i (A_L)_i \leq C_L. \end{aligned} \tag{4.3.7}$$

Here, $\mathbf{y} \in I\!R^m$ is the variable. The dual problem is interesting because it provides a machinery to formulate many problems in a natural manner.

In this context, we may drop the third part of the constraints in (4.3.6) and its dual (4.3.7), since we do not have explicit linear constraints. In the remaining of this chapter we discuss different ways to put Problem (1.3.1) in the form of (4.3.7). As a matter of fact, we can do that in three different ways depending on how we define the Frobenius norm $\|F - H\|_F$.

4.3.1 Formulation IV (SQV)

One way to define $\|F - H\|_F$ is

$$\|F - H\|_F = \|\mathbf{vec}(F - H)\|_2.$$

So, if we put $\|F - H\|_F \leq t$ for $t \in I\!R^+$, then by the definition of the second-order cone, we have

$$\begin{bmatrix} t \\ \mathbf{vec}(F - H) \end{bmatrix} \in \mathcal{Q}_{1+n^2}.$$

Hence, we have the following reformulation of (1.3.1):

$$\begin{aligned} (SQV) \quad \min \quad & t \\ \text{s.t.} \quad & \begin{bmatrix} t & 0 \\ 0 & H \end{bmatrix} \succeq 0, \\ & \begin{bmatrix} t \\ \mathbf{vec}(F - H) \end{bmatrix} \geq_Q 0. \end{aligned} \tag{4.3.8}$$

This problem is in the form of (4.3.7) with

$$\mathbf{b} = [-1\ 0\ \cdots\ 0]^T,$$

$$(A_S)_1 = \begin{bmatrix} -1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad (A_S)_k = \begin{bmatrix} 0 & 0 \\ 0 & -E_{k-1} \end{bmatrix}, \quad k = 2, \cdots, 2n-1,$$

$$(A_Q)_1 = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad (A_Q)_k = \begin{bmatrix} 0 \\ \mathbf{vec}(E_{k-1}) \end{bmatrix}, \quad k = 2, \cdots, 2n-1,$$

$$C_S = 0_{(n+1)\times(n+1)}, \qquad C_Q = \begin{bmatrix} 0 \\ \mathbf{vec}(F) \end{bmatrix},$$

where the matrices E_{k-1}'s are defined in (2.4.7), page 22. Although this formulation is natural and straightforward, we notice that the dimension of the second-order cone constraint is large, $1+n^2$. Also, in this formulation the special structure of Problem (1.3.1) is not fully exploited. Hence we should look for another way of formulation which exploits the structure and has a dimension of less magnitude. The SDP part of any mixed formulation will be the same as above. However, the second-order cone part will make the difference; since it is what we can manipulate.

4.3.2 Formulation V (SQQ)

The second definition is as introduced in Subsection 3.3.3, *i. e.* ,

$$\|F - H\|_F^2 \quad = \mathbf{y}^T P \mathbf{y} + 2\mathbf{q}^T \mathbf{y} + r$$

Hence, we have the following equivalent problem to (1.3.1)

$$\begin{array}{lll} \min & \mathbf{y}^T P \mathbf{y} + 2\mathbf{q}^T \mathbf{y} + r & \\ \text{s.t.} & H \in \mathcal{H}_n, & (4.3.9) \\ & H \succeq 0. & \end{array}$$

But

$$\mathbf{y}^T P \mathbf{y} + 2\mathbf{q}^T \mathbf{y} + r = \|P^{1/2}\mathbf{y} + P^{-1/2}\mathbf{q}\|_2^2 + r - \mathbf{q}^T P^{-1} \mathbf{q}.$$

Now, we minimize $\|F - H\|_F^2$ by minimizing $\|P^{1/2}\mathbf{y} + P^{-1/2}\mathbf{q}\|_2$. Thus we have the following problem:

$$\begin{array}{lll} (SQQ) \quad \min & t & \\ & \text{s.t.} \quad \begin{bmatrix} t & 0 \\ 0 & H \end{bmatrix} \succeq 0 & \\ & \begin{bmatrix} t \\ P^{1/2}\mathbf{y} + P^{-1/2}\mathbf{q} \end{bmatrix} \geq_Q 0, & (4.3.10) \end{array}$$

where $t \in I\!R^+$ is as before. Again, this problem is in the form of problem (4.3.7). Here, the difference between this form and SQV is in the second-order cone

constraint since the SDP part is the same as SQV. Indeed, we have

$$(A_Q)_1 = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (A_Q)_k = \begin{bmatrix} 0 \\ -\mathbf{p}_{k-1} \end{bmatrix}, \quad k = 2, \cdots, 2n-1, \quad C_Q = \begin{bmatrix} 0 \\ P^{-1/2}\mathbf{q} \end{bmatrix}.$$

The j^{th} component of $\mathbf{p}_{k-1} \in I\!R^{2n-1}$ is

$$(\mathbf{p}_{k-1})_j = \begin{cases} \sqrt{j} & \text{if } \ j = k-1 \ \text{ and } \ k-1 \leq n, \\ \sqrt{2n-j} & \text{if } \ j = k-1 \ \text{ and } \ k-1 > n, \\ 0 & \text{otherwise.} \end{cases}$$

The dimension of the second-order cone in SQV is $1+n^2$ and in SQQ is just $2n$, which makes us expect less efficiency in practice when we work with SQV. The optimal value of SQV is the same as that of problem (1.3.1), whereas the optimal values of SQQ and (1.3.1) are equal up to a constant. Indeed, the optimal value of (4.3.9) is equal $(\rho^*)^2 + r - \mathbf{q}^T P^{-1}\mathbf{q}$, where ρ^* is the optimal value of (4.3.10). One might notice that we did not talk about the constraint of H being Hankel. This is because the Hankel structure of H is embedded in the other constraints.

4.3.3 Formulation VI (SQH)

The last formulation will take advantage of the Hankel structure of H explicitly. The more economical vectorization operator **hvec** on Hankel matrices, introduced

in Section 3.3.2 will be used. From Theorem 3.3.2, we have the following:

$$\|\hat{F} - H\|_F = \|\mathbf{hvec}(\hat{F} - H)\|_2,$$

where $\hat{F} = P_H(F)$, so that we have the following problem:

$$\begin{array}{lll} (SQH) & \min & t \\ & \text{s.t.} & \begin{bmatrix} t & 0 \\ 0 & H \end{bmatrix} \succeq 0 \\ & & \begin{bmatrix} t \\ \mathbf{hvec}(\hat{F} - H) \end{bmatrix} \geq_Q 0. \end{array} \tag{4.3.11}$$

The dimension of the second-order cone in this form is $2n$, the same as that of SQQ. More precisely, we have

$$(A_Q)_1 = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad (A_Q)_k = \begin{bmatrix} 0 \\ \mathbf{hvec}(E)_{k-1} \end{bmatrix}, \quad k = 2, \cdots, 2n-1,$$

$$C_Q = \begin{bmatrix} 0 \\ \mathbf{hvec}(\hat{F}) \end{bmatrix}. \quad E_{k-1}\text{'s are the same as above}$$

Furthermore, the optimal value is the same as that of (1.3.1).

Table 4.1 shows the dimensions of the semidefinite part (SD part) and the second-order cone part (SOC part) for each formulation. For the formulations SDH and SDQ, the second-order cone part is not applicable, so the cell in the

Formulation	SD part	SOC part
SDV	$2n \times (n^2 + n + 2)$	
SDH	$2n \times (3n + 1)$	
SDQ	$2n \times (3n + 1)$	
SQV	$2n \times (n + 1)$	$n^2 + 1$
SQQ	$2n \times (n + 1)$	$2n$
SQH	$2n \times (n + 1)$	$2n$

Table 4.1: Problem dimensions

table corresponding to that is left blank.

In practice, we expect that the mixed formulations are more efficient than the SDP-only formulations, especially the SQQ and SQH which have second-order cone constraint of least dimension. Since interior point methods for SOCP have better worst-case complexity than an SDP method. However, the formulation SDH has a less SDP dimension with no illness such as SDQ has, which makes SDH a better choice among other SDP-only formulations. This is due to the economical vectorization operator **hvec**. Indeed, practical experiments show a competitive behavior of SDH to SQQ and SQH (see Chapter 6).

Chapter 5

Interior Point Methods

5.1 Preamble

The field of *interior point methods* for LP more or less started with ellipsoid algorithm of Khachiyan [27], that allowed a polynomial bound on the worst-case iteration count. This resolved the question whether linear programming problems are solvable in polynomial time. In practice, however, the ellipsoid method is slow. The next major development was the famous paper by Karmarkar [26] in 1984, who introduced an algorithm with an improved complexity bound that was also accompanied by claims of computational efficiency. Karmarkar's method is considered the actual start of interior point methods. Although, the prehistory of this subject started in 1967 when Dikin[1] invented what is now called the affine scaling algorithm for LP, hence interior point methods exist at least since 1967. The reasons why this method was overlooked may be: first, Dikin came too early, when there was no interest in iterative procedures for LP. Second, at that time there were neither hardware base for Dikin to perform large-scale tests of his algorithm nor social demand for solving large-scale LP problems.

In the following decade, after Karmarkar, several thousands papers appeared on this topic. One of these papers, which may be considered another cornerstone

[1] I found this remark in [34]. Unfortunately, I could not find the original paper.

in the development of interior point methods, is due to Renegar[38] where the first path-following polynomial time interior point method for LP was developed.

An important breakthrough was achieved by Nesterov and Nemirovsky in 1988 [34]. They showed that interior point methods for LP can, in principle, be generalized to all convex optimization problems. The key element is the knowledge of a barrier function with a certain property: self-concordance. To be useful in practice, the barrier (or really, its first and second derivatives) must also be computable. They also show that a self-concordant function exists for every convex set, but unfortunately their universal self-concordant barrier is not readily computable. Independently of Nesterov and Nemirovsky, Alizadeh [4] generalized interior point methods from linear programming to semidefinite programming. Initially, research papers gave the impression that extension of interior point methodology to semidefinite programming was rather straightforward. The current insight, however, is that a substantial research effort on interior point methods for semidefinite programming is still necessary. One of the first surprises came when several research groups each introduced quite different generalization of the primal-dual interior point method to semidefinite programming. In particular, Alizadeh et. al. [7] introduced the AHO direction, Helmberg et. al. [23] introduced the HKM direction, and Nesterov and Todd [35, 36] introduced the NT direction.

In an attempt to extend the primal-dual approach beyond semidefinite programming, Nesterov and Todd [35, 36] introduced the concept of self-scaled cones. Their work can be considered as the first papers dealing with mixed semidefinite and second-order cone optimization problems.

There are other approaches to solve SDP than the primal-dual interior point methods. Such approaches include dual-only interior point methods, bundle methods, augmented Lagrangian methods, non-smooth Newton methods, among others.

One class of interior point methods is the *primal-dual path-following methods*. These methods are carefully named: the underlying idea for these algorithms is to 'follow' a 'path' approximately in order to reach the optimal set. These methods conform to the general scheme of Nesterov and Nemirovsky [34], where they were first introduced (for SDP) and analyzed. For up to date announcements regarding interior point methods one may sign up in the web site [53] for the interior point mailing list.

In this chapter, we will give a brief description of the primal-dual path-following method used to solve SDP problems. This method differs from the other version applied onto mixed problems technically. However, they all have the same general scheme. The reason for choosing the SDP version of the path-following method is due to its simplicity in terms of notation and in terms of demonstrating the method. Furthermore, dealing with second-order cone constraints explicitly needs some knowledge of the theory of Euclidean Jordan algebra. This theory provides an approach to study SDP and SOCP in a unified way (see, [41]). Introducing this theory is not really necessary to understand the fundamental concepts of the path-following algorithm which are the same for SDP, SOCP and a mixture of them.

In Section 2 we give a brief review of the theoretical properties of SDP concerning duality and optimality conditions. We define what is called the *central path* with existence and uniqueness theorems in Section 3. A generic scheme of primal-dual path-following interior point methods is presented in Section 4. We conclude this chapter by demonstrating the method by solving a simple example. That will be done in Section 5.

5.2 Duality and Optimality

We study the SDP in primal form (3.2.1) and its dual (3.2.2), *i.e.* ,

$$\begin{aligned} (P) \quad & \min_X C \bullet X \\ & \text{s. t. } A_i \bullet X = b_i, \quad i = 1, \cdots, m, \\ & X \succeq 0, \end{aligned}$$

and

$$\begin{aligned} (D) \quad & \max_{\mathbf{y}} \mathbf{b}^T \mathbf{y} \\ & \text{s. t.} \\ & \sum_{i=1}^{m} y_i A_i + S = C. \\ & S \succeq 0, \end{aligned}$$

We write $\mathcal{P}$ and $\mathcal{D}$ for the *primal* and the *dual feasible sets*. Hence,

$$\mathcal{P} = \{X \in \mathcal{S}_n^+ : A_i \bullet X = b_i \ (i = 1, \cdots, m)\},$$

and

$$\mathcal{D} = \{(\mathbf{y}, S) \in I\!R^m \times \mathcal{S}_n^+ : \sum_{i=1}^{m} y_i A_i + S = C\}$$

X and $(\mathbf{y}, S)$ are *feasible solutions* if $X \in \mathcal{P}$ and $(\mathbf{y}, S) \in \mathcal{D}$. We also denote the interior of $\mathcal{P}$ and $\mathcal{D}$ by $\mathcal{P}^+$ and $\mathcal{D}^+$, respectively. The following assumptions will be made throughout this chapter:

A.1 The matrices A_i $(i = 1, \cdots, m)$ are linearly independent.

A.2 (Strict feasibility) The sets $\mathcal{P}^+$ and $\mathcal{D}^+$ are nonempty.

The difference between the primal and the dual objective values at feasible solutions of (P) and (D) is called the *duality gap.*

Definition 5.2.1 (Duality gap)

Let $X \in \mathcal{P}$ and $(\mathbf{y}, S) \in \mathcal{D}$. The quantity

$$C \bullet X - \boldsymbol{b}^T \boldsymbol{y}$$

is called the duality gap of (P) and (D) at $(X, \mathbf{y}, S)$. △

Theorem 5.2.1 (Weak Duality)

Let $X \in \mathcal{P}$ and $(\mathbf{y}, S) \in \mathcal{D}$. Then

$$C \bullet X - \boldsymbol{b}^T \boldsymbol{y} = S \bullet X \geq 0,$$

i.e. , the duality gap is nonnegative at feasible solutions. □

Proof:

For feasible solutions $X \in \mathcal{P}$ and $(\mathbf{y}, S) \in \mathcal{D}$, we have

$$\begin{aligned} C \bullet X - \mathbf{b}^T\mathbf{y} &= \mathbf{trace}(CX) - \mathbf{b}^T\mathbf{y} \\ &= \mathbf{trace}\left(\left(\sum_{i=1}^{m} y_i A_i + S\right) X\right) - \sum_{i=1}^{m} y_i A_i \bullet X \\ &= \mathbf{trace}(SX) = S \bullet X, \end{aligned}$$

But, X and $S \in \mathcal{S}_n^+$, so

$$\begin{aligned} S \bullet X = \mathbf{trace}(SX) &= \mathbf{trace}(S^{1/2} S^{1/2} X^{1/2} X^{1/2}) \\ &= \mathbf{trace}(S^{1/2} X^{1/2} X^{1/2} S^{1/2}) \\ &= \|S^{1/2} X^{1/2}\|_F^2 \geq 0. \end{aligned}$$

and this completes the proof ■

A strong duality theorem is available as well.

Theorem 5.2.2 (Strong Duality)

The existence of strictly feasible solutions to (P) and (D) implies that both have bounded nonempty optimal solution sets, with zero duality gap. □

Proof:(see [52], page 276). ■

In order to get necessary and sufficient optimality conditions, we introduce the following lemma.

Lemma 5.2.1 If $X \in \mathcal{S}_n^+$ and $S \in \mathcal{S}_n^+$ and $X \bullet S = 0$, then $XS = SX = 0$

Proof:

$$\begin{aligned} S \bullet X = \mathbf{trace}(SX) &= \mathbf{trace}(S^{1/2}S^{1/2}X^{1/2}X^{1/2}) \\ &= \mathbf{trace}(S^{1/2}X^{1/2}X^{1/2}S^{1/2}) \\ &= \|S^{1/2}X^{1/2}\|^2. \end{aligned}$$

Thus if $S \bullet X = 0$, it follows that $S^{1/2}X^{1/2} = 0$. Pre-multiplying by $S^{1/2}$ and post-multiplying by $X^{1/2}$ yields $SX = 0$, which in turn implies $(SX)^T = XS = 0$. ■

Now, by weak duality (Theorem 5.2.1) we know that $X^* \in \mathcal{P}$ and $(\mathbf{y}^*, S^*) \in \mathcal{D}$ will be optimal if the duality gap at $(X^*, \mathbf{y}^*, S^*)$ is zero, *i.e.* $X^* \bullet S^* = 0$. Hence, by Lemma 5.2.1, $X^*S^* = 0$. It follows that the following conditions (together with $X \in \mathcal{S}_n^+$ and $S \in \mathcal{S}_n^+$) are sufficient for X and $(\mathbf{y}, S)$ to be optimal solutions:

$$\left.\begin{array}{rl} A_i \bullet X = b_i, & i = 1, \cdots, m \\ \sum_{i=1}^m y_i A_i + S = C, & \\ XS = 0. & \end{array}\right\} \qquad (5.2.1)$$

The condition $XS = 0$ is called the *complementarity condition*, and optimal solutions that satisfy this condition are called *complementary*.

The strong duality (Theorem 5.2.2) implies that these optimality conditions are also necessary if (P) and (D) are strictly feasible.

Theorem 5.2.3 (Necessary and sufficient optimality conditions)

Under Assumption A.2 (strict feasibility), (5.2.1) is a system of necessary and sufficient optimality conditions for (P) and (D). □

Let us simplify the notation slightly: we define the operator $\mathcal{A} : \mathcal{S}_n \to I\!R^m$ by

$$(\mathcal{A}X)_i := A_i \bullet X, \quad i = 1, \cdots, m.$$

Then the adjoint[2] of this operator is $\mathcal{A}^* : I\!R^m \to \mathcal{S}_n$ satisfying

$$\mathcal{A}^*\mathbf{y} = \sum_{i=1}^{m} y_i A_i.$$

Using this notation, we can rewrite the above equations as

$$\left.\begin{array}{rrl} \mathcal{A}X & & = \mathbf{b}, \\ \mathcal{A}^*\mathbf{y} + & S & = C, \\ XS & & = 0. \end{array}\right\} \tag{5.2.2}$$

If the system of necessary and sufficient optimality conditions for (P) and (D) is perturbed by introducing a parameter $\mu > 0$ in a special way, then the solution of the perturbed system defines an analytic curve (parameterized by μ) through the feasible region, which leads to the optimal set as $\mu \downarrow 0$. This curve is called the central path and most interior point methods 'follow' the *central path* approximately to reach the optimal set.

[2]The operator $T^* : Y \to X$ is called the adjoint operator of the operator $T : X \to Y$ if $< T(x), y > = < x, T^*(y) > \forall \ x \in X$ and $y \in Y$ where $< \cdot, \cdot >$ is an appropriate inner product.

5.3 The Central Path

We now perturb the optimality conditions (5.2.2) for (P) and (D) to

$$\left.\begin{aligned} \mathcal{A}X &= \mathbf{b}, \\ \mathcal{A}^*\mathbf{y} + S &= C, \\ XS &= \mu I. \end{aligned}\right\} \tag{5.3.3}$$

for some $\mu > 0$. Note that if $\mu = 0$ we regain the optimality conditions (5.2.2).

Definition 5.3.1 (Central Path)

The central path is defined as the set of solutions $(X, \boldsymbol{y}, S) = (X(\mu), \boldsymbol{y}(\mu), S(\mu)) \in \mathcal{S}_n^+ \times I\!R^m \times \mathcal{S}_n^+$ *to (5.3.3), for all* $\mu > 0$. △

Clearly any solution to equations (5.3.3) gives strictly feasible solution to both (P) and (D), since the last condition implies that X and S are nonsingular, hence positive definite.

The following theorem shows that the existence of strictly feasible solutions to both problems in also sufficient for the existence and uniqueness of solutions to (5.3.3) for every $\mu > 0$.

Theorem 5.3.1

Suppose that both (P) and (D) have strictly feasible solutions. Then, for every $\mu > 0$, *there exits a unique solution* $(X(\mu), \boldsymbol{y}(\mu), S(\mu))$ *in* $\mathcal{S}_n^+ \times I\!R^m \times \mathcal{S}_n^+$ *to the central path equations (5.3.3).*

Proof:(See, [52], page 274). ■

Theorem 5.3.1 proved the existence and uniqueness of points on the central path, but we have not justified calling it a path. This will follow if we show that the equations defining it are differentiable, with a derivative that is square and nonsingular at points on the path. Unfortunately, while the equations (5.3.3) are differentiable, the derivative is not even square since the left-hand side maps $(X, y, S) \in \mathcal{S}_n^+ \times I\!R^m \times \mathcal{S}_n^+$ to a point in $\mathcal{S}_n \times I\!R^m \times I\!R^{n\times n}$. Because XS is usually not symmetric even if X and S are. We therefore need to change the equations defining the central path. There are many possible approaches which lead to different search directions for path-following algorithms, but for now we choose a simple one. We replace $XS = \mu I$ by $-\mu X^{-1} + S = 0$. It can be shown that the function $X \to -\mu X^{-1}$ is differentiable (see [13], page 244), with derivative $\mu(X^{-1} \odot X^{-1})$. Here the notation $Q \odot R$ for an $n \times n$ matrices Q and R(usually symmetric) is used for an operator from $\mathcal{S}_n$ to $\mathcal{S}_n$ which is defined as

$$(Q \odot R)U := \frac{1}{2}(QUR^T + RUQ^T).$$

So the central path is defined by the equations:

$$\Phi_P(X, \mathbf{y}, S) := \begin{bmatrix} \mathcal{A}X & & \\ & \mathcal{A}^*\mathbf{y} & + S \\ -\mu X^{-1} & & + S \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ C \\ \mathbf{0} \end{bmatrix}$$

whose derivative is

$$\Phi'_P(X, \mathbf{y}, S) := \begin{bmatrix} \mathcal{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathcal{A}^* & \mathcal{I} \\ \mu(X^{-1} \odot X^{-1}) & \mathbf{0} & \mathcal{I} \end{bmatrix}$$

where $\mathcal{I}$ denotes the identity operator on $\mathcal{S}_n$. The blocks are operators not matrices. We want to show that this derivative is nonsingular, and for this it suffices to prove that its null-space is trivial. We derive this from a more general result.

Theorem 5.3.2

Suppose the operators $\mathcal{E}$ and $\mathcal{F}$ map $\mathcal{S}_n$ to itself, and that $\mathcal{E}$ is nonsingular and $\mathcal{E}^{-1}\mathcal{F}$ is positive definite. Then the solution to

$$\left.\begin{array}{rcrl} \mathcal{A}x & & & = r_p, \\ & \mathcal{A}^*y + & s & = R_d, \\ \mathcal{E}x & + & \mathcal{F}s & = R_c. \end{array}\right\} \tag{5.3.4}$$

is uniquely given by

$$\begin{aligned} y &= (\mathcal{A}\mathcal{E}^{-1}\mathcal{F}\mathcal{A}^*)^{-1}(r_p - \mathcal{A}\mathcal{E}^{-1}(R_c - \mathcal{F}R_d)), \\ s &= R_d - \mathcal{A}^*y, \\ x &= \mathcal{E}^{-1}(R_c - \mathcal{F}s), \end{aligned}$$

where $(x, y, s) \in (\mathcal{S}_n, I\!R^m, \mathcal{S}_n)$. □

Proof:

The formulae for s and x follow directly from the second and the third equations. Substituting for s in the formula for x, and inserting this in the first equation, we obtain after some manipulations

$$(\mathcal{A}\mathcal{E}^{-1}\mathcal{F}\mathcal{A}^*)y = r_p - \mathcal{A}\mathcal{E}^{-1}(R_c - \mathcal{F}R_d).$$

Since $\mathcal{E}^{-1}\mathcal{F}$ is positive definite and by Assumption A.1 that A_i's are linearly independent, therefore the $m \times m$ matrix on the left is positive definite and hence nonsingular. This verifies that y is uniquely determined as given, and then so are s and x. Moreover, these values solve the equations. ■

In the previous discussion we symmetrize the third equation of (5.3.3) by replacing it by $-\mu X^{-1} + S = 0$. A more general approach of symmetrizing is to replace $XS = \mu I$ by

$$\mathcal{T}_P(XS) = \mu I,$$

where $\mathcal{T}_P$ is the linear operator given by

$$\mathcal{T}_P(U) := \frac{1}{2}[PUP^{-1} + P^{-T}U^T P^T],$$

for any matrix U, and where the matrix P determines the symmetrization strategy. Below are some popular choices for P.

(1) $P = I$ corresponds to the AHO direction [7].

(2) $P = S^{1/2}$ corresponds to the HKM direction [23, 29, 33].

(3) $\left[X^{1/2}\left(X^{1/2}SX^{1/2}\right)^{-1/2}X^{1/2}\right]^{1/2}$ corresponds to the NT direction [46].

Primal-dual path-following methods solve (5.3.3) approximately, followed by a reduction in μ. The goal is to obtain primal and dual directions ΔX and $(\Delta\mathbf{y}, \Delta S)$, respectively, that satisfy $X + \Delta X \succeq 0$, $\mathbf{y} + \Delta\mathbf{y} \geq 0$ and $S + \Delta S \succeq 0$. Hence, the computation of $(\Delta X, \Delta\mathbf{y}, \Delta S)$ requires to solve the system (5.3.3) for $(X + \Delta X, \mathbf{y} + \Delta\mathbf{y}, S + \Delta S)$ after symmetrizing with $\mathcal{T}_P$ (with respect to an invertible P). This can be done by solving the symmetrized Newton equation

given by

$$\left.\begin{array}{rcrcll} \mathcal{A}\Delta X & & & = r_p & := \mathbf{b} - \mathcal{A}X, \\ & \mathcal{A}^*\Delta\mathbf{y} & + \quad \Delta S & = R_d & := C - S - \mathcal{A}^*\mathbf{y}, \\ \mathcal{E}\Delta X & & + \quad \mathcal{F}\Delta S & = R_c & := \sigma\mu I - \mathcal{T}(XS). \end{array}\right\} \tag{5.3.5}$$

Here σ is a centering parameter. $\mathcal{T}_P$ is the symmetrization operator defined above and $\mathcal{E}$ and $\mathcal{F}$ are the linear operators

$$\mathcal{E} = P \circledast P^{-T}S, \quad \mathcal{F} = PX \circledast P^{-T}, \tag{5.3.6}$$

where $Q \circledast R$ denotes the linear operator defined by

$$\begin{aligned} &Q \circledast R : \mathcal{S}_n \longrightarrow \mathcal{S}_n \\ &Q \circledast R(U) = \tfrac{1}{2}[QUR^T + RUQ^T]. \end{aligned}$$

Now, since P is invertible therefore $\mathcal{E}$ is nonsingular and $\mathcal{E}^{-1}\mathcal{F}$ is positive definite, and by Theorem 5.3.2 we have

$$M\Delta\mathbf{y} = h, \tag{5.3.7}$$

where

$$M = \mathcal{A}\mathcal{E}^{-1}\mathcal{F}\mathcal{A}^*, \tag{5.3.8}$$

$$h = r_p - \mathcal{A}\mathcal{E}^{-1}(R_c - \mathcal{F}R_d), \tag{5.3.9}$$

Then we compute ΔX and ΔS from the equations

$$\Delta S = R_d - \mathcal{A}^* \Delta \mathbf{y}, \tag{5.3.10}$$

$$\Delta X = \mathcal{E}^{-1}(R_c - \mathcal{F}s). \tag{5.3.11}$$

The most expensive step in each iteration of a path-following algorithm is the computation of the matrix M (for example, the implementation SDPT3 , which we will use in our numerical results, spent from 50% to 80% of the total CPU time in computing M). From equation (5.3.8), it is easily shown that the (i, j) element of M is given by

$$M_{ij} = A_i \bullet \mathcal{E}^{-1}\mathcal{F}(A_j). \tag{5.3.12}$$

Thus for a fixed j, computing first the matrix $\mathcal{E}^{-1}\mathcal{F}(A_j)$ and then taking its product with each A_i, $i = 1, \cdots, m$, gives the jth column of M.

5.3.1 The Generic Algorithm

We are now ready to introduce a general scheme of the path-following method.

Algorithm 5.3.1 (Generic primal-dual path-following algorithm)

Input

An initial iterate $(X^0, \mathbf{y}^0, S^0) \in \mathcal{P} \times \mathcal{D}$, with $X^0 \succ 0$ and $S^0 \succ 0$.

Parameters

An invertible matrix P to decide on the symmetrization method,

An accuracy parameter $\epsilon > 0$

Begin

Set $(X, \mathbf{y}, S) := (X^0, \mathbf{y}^0, S^0)$, and $\mu = (X \bullet S)/n$

while $\mu > \epsilon$ **do**

Compute $(\Delta X, \Delta \mathbf{y}, \Delta S)$ from (5.3.7), (5.3.10) and (5.3.11).

Choose a step length $\alpha \in (0, 1]$.

Update:

$X := X + \alpha \Delta X$,

$\mathbf{y} := \mathbf{y} + \alpha \Delta \mathbf{y}$,

$S := S + \alpha \Delta S$,

$\mu := (X \bullet S)/n$.

end

end

This algorithm is very general, so we need some specifications to make it work practically. In what remains of this section we present some implementational remarks. These remarks coincide with the software SDPT3.

Remarks:

- The algorithm starts with a strictly feasible iterate $(X^0, \mathbf{y}^0, S^0)$, hence this algorithm is sometimes called feasible primal-dual path-following algorithm. However, this algorithm may start with infeasible iterate and then use, for example, the traditional big-M strategy to get a feasible start. An elegant way of avoiding the big-M method is to embed the SDP problem in a larger problem that is essentially its own dual, and for which a feasible solution is known. A solution of this self-dual embedding problem then gives information about the solution of the original problem.

- To see how we can compute the step-length α, suppose we have $X \succ 0$ as the current primal iterate, and we want to find a step-length α such that $X + \alpha\Delta X \succeq 0$. To do this let $\alpha_{\max}$ denote the maximum allowed step-length. $X + \alpha\Delta X \succeq 0$ is equivalent to $I + \alpha X^{-1}\Delta X \succeq 0$. In other words, we should have $1 + \alpha\lambda_{\min}(X^{-1}\Delta X) \geq 0$, where $\lambda_{\min}$ is the minimum eigenvalue of $X^{-1}\Delta X$. Thus

$$\alpha_{\max} = \begin{cases} \frac{-1}{\lambda_{\min}} & \text{if } \lambda_{\min} < 0, \\ \infty & \text{otherwise.} \end{cases}$$

  Once the maximum allowed step-length is determined, the next primal iterate is taken to be

$$X_+ = X + \alpha\Delta X,$$

  where $\alpha = \min(1, \gamma\alpha_{\max})$ and γ is a positive parameter that is usually set at a value slightly smaller than 1 (say, 0.98) to prevent the new iterate from getting too close to the boundary of the cone of positive semidefinite

matrices. The same argument applies for the dual iterate, hence we have

$$\mathbf{y}_+ = \mathbf{y} + \beta\Delta\mathbf{y}, \quad S_+ = S + \beta\Delta S.$$

where $\beta = \min(1, \gamma\beta_{\max})$ and

$$\beta_{\max} = \begin{cases} \frac{-1}{\lambda_{\min}(S^{-1}\Delta S)} & \text{if } \lambda_{\min}(S^{-1}\Delta S) < 0, \\ \infty & \text{otherwise.} \end{cases}$$

Toh [48] discusses all known possible methods to compute the step-lengths in interior point methods for SDP problems.

- The centering parameter σ associated with the complementarity condition in (5.3.5) can be chosen for the next iteration so as to get as much reduction in the total complementarity $X \bullet S$ (the duality gap) as possible. Thus it can be updated in each iteration depending on the current step-lengths. Indeed, the next value of this parameter is given by

$$\sigma_+ = 1 - 0.9\min(\alpha, \beta).$$

- The algorithm stops when both of the following criteria are less than a given tolerance ϵ:
  - The duality gap $X \bullet S$, and

– an infeasibility measure ϕ which can be defined as

$$\phi = \max\left(\frac{\|r_p\|}{\max(1,\|b\|)}, \frac{\|R_d\|_F}{\max(1,\|C\|_F)}\right). \tag{5.3.13}$$

There are more stopping criteria, but these two are enough for our purpose.

- Algorithms differ in how μ is updated. Methods that use large reduction of μ followed by damped step-lengths are called long-step methods. Methods that use dynamic updates of μ include the popular predictor-corrector methods (see [7]).

5.4 Illustrative Example

To demonstrate Algorithm 5.3.1, bearing in mind the aforementioned remarks, we consider a simple LP problem after casting it as an SDP problem. We consider such a simple example for presentation purposes and also the ability to visualize the mechanism of the method.

Example

Consider the following LP problem:

$$\begin{aligned} \min \quad & x_1 - x_2 \\ \text{s.t.} \quad & \\ & x_1 - x_2 \leq 2 \\ & x_1 + x_2 \leq 6 \\ & x_1 \;,\; x_2 \geq 0 \end{aligned} \tag{5.4.14}$$

The solution will be organized in the following steps:

Formulation

Introducing slack variables $x_3,\ x_4 \geq 0$, and then casting (5.4.14) as an SDP in the primal form gives

$$\begin{aligned} \min \quad & C \bullet X \\ \text{s.t.} \quad & \\ & A_1 \bullet X = b_1 \\ & A_2 \bullet X = b_2 \\ & X \succeq 0 \end{aligned} \tag{5.4.15}$$

where

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 6 \end{bmatrix},$$

and

$$X = \begin{bmatrix} x_1 & 0 & 0 & 0 \\ 0 & x_2 & 0 & 0 \\ 0 & 0 & x_3 & 0 \\ 0 & 0 & 0 & x_4 \end{bmatrix}$$

Initial settings

We start the primal-dual path-following algorithm with the following initial iterate:

$$X^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}, \mathbf{y}^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, S^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

which is strictly feasible to the primal and the dual problems. Also, we use the AHO direction with symmetrizing matrix $P = I_{4\times 4}$(the identity), for simplicity and convenience. Hence, the operators $\mathcal{T}_P(\cdot)$, $\mathcal{E}$ and $\mathcal{F}$ are defined as

$$\left.\begin{array}{l} \mathcal{T}_I(U) = \frac{1}{2}[U + U^T], \\ \mathcal{E}(V) = (I \circledast S)(V) = \frac{1}{2}[VS + SV], \\ \mathcal{F}(V) = (X \circledast I)(V) = \frac{1}{2}[XV + VX]. \end{array}\right\} \quad \forall U \in I\!R^{4\times 4}, \ V \in \mathcal{S}_4$$

The parameter $\gamma = 0.9$ will be fixed throughout the process of the algorithm. Finally we set $\sigma^0 = 0.5$ and update it at each iteration. In the following we will describe the first iteration in detail, and give the results for the last iteration when the stopping criterion is met with tolerance $\epsilon = 10^{-4}$.

The first iteration

- We start with computing $\mu^0 = \frac{X^0 \bullet S^0}{4} = 2.5$

- We compute the infeasibility quantities r_p, R_d and the complimentarily R_c.

$$\mathcal{A}X^0 = \begin{bmatrix} A_1 \bullet X^0 \\ A_2 \bullet X^0 \end{bmatrix} = \begin{bmatrix} 2 \\ 6 \end{bmatrix}, \mathcal{A}^*(\mathbf{y}^0) = y_1^0 A_1 + y_2^0 A_2 = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathcal{T}_I(X^0 S^0) = \tfrac{1}{2}[X^0 S^0 + S^0 X^0] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}.$$

hence,

$$r_p = \mathbf{b} - \mathcal{A}X^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$R_d = C - S^0 - \mathcal{A}^*(\mathbf{y}^0) = \begin{bmatrix} -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix},$$

$$R_c = \sigma^0 \mu^0 I - \mathcal{T}_I(X^0 S^0) = \begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0 & -1.75 & 0 & 0 \\ 0 & 0 & -2.75 & 0 \\ 0 & 0 & 0 & -0.75 \end{bmatrix}$$

- We check, now, for infeasibility by evaluating ϕ from formula (5.3.13) to get $\phi = 2.8284 > \epsilon$. And we check for optimality by evaluating the duality gap $X^0 \bullet S^0/4 = 10 > \epsilon$; so continue.

- Now, we construct the matrix M. To do so
  - We compute M_{ij} by first computing

$$\mathcal{F}(A_1) = \frac{1}{2}[X^0 A_1 + A_1 X^0] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

  - Then we compute $\mathcal{E}^{-1}\mathcal{F}(A_1)$. We have here a problem since we do not have an explicit formula for $\mathcal{E}^{-1}$. So we get around this by letting $\mathcal{E}^{-1}(Z) = W$, where $Z = \mathcal{F}(A_1)$. Thus, we have $\mathcal{E}(W) = Z$, or equivalently

$$WS^0 + S^0 W = 2Z$$

    which is known as a *Lyapunov system*, so we need to solve this system for W. In this case S^0 is diagonal. Hence,

$$\begin{array}{rccccc} & WS^0 & + & S^0 W & = & 2Z \\ \Leftrightarrow & S_{jj}W_{ij} & + & S_{ii}W_{ij} & = & 2Z_{ij} \\ \Leftrightarrow & & & W_{ij} & = & \frac{2Z_{ij}}{S_{ii}+S_{jj}}. \end{array}$$

    Therefore,

$$\mathcal{E}^{-1}\mathcal{F}(A_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

– and

$$M_{11} = A_1 \bullet \mathcal{E}^{-1}\mathcal{F}(A_1) = 8.$$

– Applying the same strategy we obtain

$$M = \begin{bmatrix} 8 & -2 \\ -2 & 6 \end{bmatrix}.$$

- Next, we form h from equation (5.3.9).

  – We compute $\mathcal{F}(R_d)$. Then

  – We compute $\mathcal{E}^{-1}\mathcal{F}(R_d)$ and $\mathcal{E}^{-1}(R_c)$ which include solving two Lyapunov systems, as above. Thus we obtain

$$h = r_p + \mathcal{A}\mathcal{E}^{-1}\mathcal{F}(R_d) - \mathcal{A}\mathcal{E}^{-1}(R_c) = \begin{bmatrix} -3.25 \\ -9.75 \end{bmatrix}.$$

- We use equations (5.3.7), (5.3.10) and (5.3.11) to find $(\Delta \mathbf{y})^1$, $(\Delta S)^1$ and

$(\Delta X)^1$, respectively, as follows:

$$(\Delta \mathbf{y})^1 = M^{-1}\mathbf{y}^0 = \begin{bmatrix} -0.8864 \\ -1.9205 \end{bmatrix},$$

$$(\Delta S)^1 = R_d - \mathcal{A}^*((\Delta \mathbf{y})^1) = \begin{bmatrix} 0.8068 & 0 & 0 & 0 \\ 0 & -0.9659 & 0 & 0 \\ 0 & 0 & -1.1136 & 0 \\ 0 & 0 & 0 & -0.0795 \end{bmatrix},$$

$$(\Delta X)^1 = \mathcal{E}^{-1}R_c - \mathcal{E}^{-1}\mathcal{F}((\Delta S)^1)$$

$$= \begin{bmatrix} -0.5568 & 0 & 0 & 0 \\ 0 & 1.1477 & 0 & 0 \\ 0 & 0 & 1.7045 & 0 \\ 0 & 0 & 0 & -0.5909 \end{bmatrix}.$$

- Now, we compute the primal and dual step-lengths α^1 and β^1 as follows:

$$\alpha^1 = \min(1, \frac{-\gamma}{\lambda_{\min}((X^0)^{-1}(\Delta X)^0)}) = 1$$
$$\beta^1 = \min(1, \frac{-\gamma}{\lambda_{\min}((S^0)^{-1}(\Delta S)^0)}) = 0.8082$$

we should note here that we ignore the term if the eigenvalue of the corresponding expression is positive.

- Update

$$X^1 = X^0 + \alpha^1(\Delta X)^1 = \begin{bmatrix} 0.4432 & 0 & 0 & 0 \\ 0 & 4.1477 & 0 & 0 \\ 0 & 0 & 5.7045 & 0 \\ 0 & 0 & 0 & 1.4091 \end{bmatrix},$$

$$\mathbf{y}^1 = \mathbf{y}^0 + \beta^1(\Delta \mathbf{y})^1 = \begin{bmatrix} 0.2837 \\ -0.5520 \end{bmatrix},$$

$$S^1 = S^0 + \beta^1(\Delta S)^1 = \begin{bmatrix} 1.6520 & 0 & 0 & 0 \\ 0 & 0.2194 & 0 & 0 \\ 0 & 0 & 0.1000 & 0 \\ 0 & 0 & 0 & 0.9357 \end{bmatrix}.$$

- Update the centering parameter $\sigma^1 = 1 - 0.9\min(\alpha^1, \beta^1) = 0.2727$.

Final iteration

The iterations go smoothly in the next iterations following the same procedures in the first, until we reach the sixth iteration where we have

$$\phi^6 = 3.1402 \times 10^{-16} < \epsilon, \qquad \mu^6 = 4.5848 \times 10^{-5} < \epsilon.$$

However, $\phi^5 = 3.1402 \times 10^{-16} < \epsilon$ but $\mu^5 = 4.5681 \times 10^{-4} > \epsilon$. So the algorithm

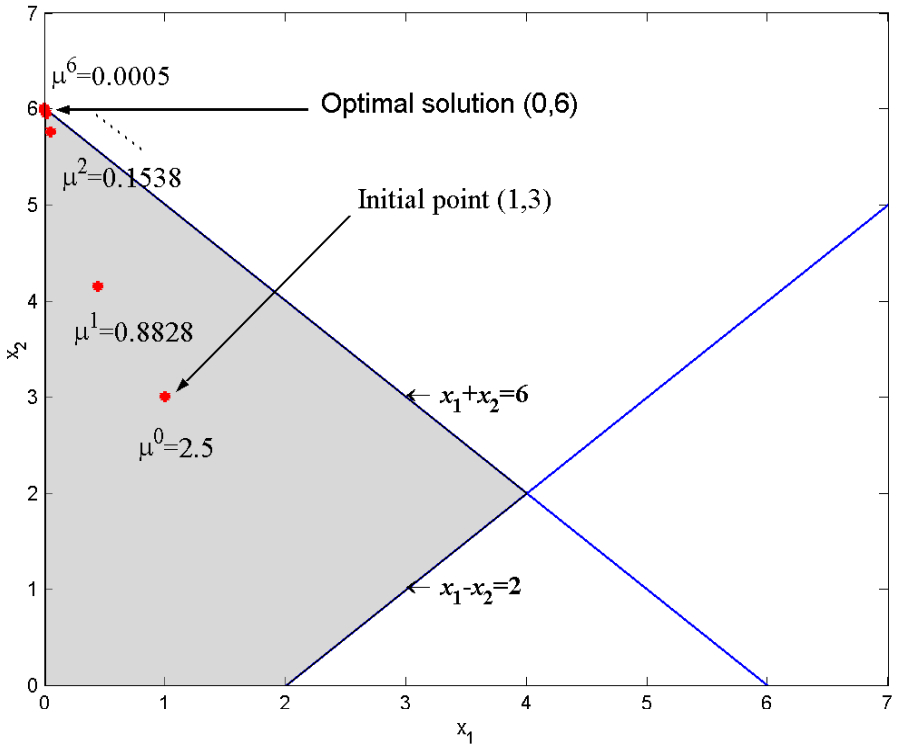

Figure 5.1: The feasible region for Problem (5.4.14).

terminates with the optimal solution

$$X^* = X^6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{y}* = \mathbf{y}^6 = \begin{bmatrix} 0 \\ -1 \end{bmatrix},$$

$$S^* = S^6 = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

which means that the optimal $\begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix} = \begin{bmatrix} 0 \\ 6 \end{bmatrix}$.

Figure 5.4 shows the feasible region of the primal problem. It also shows the progress of the algorithm with feasible points along with the corresponding values of μ to see how the algorithm converges to the optimal solution through the region.

Chapter 6

Numerical Results

We will now present some numerical results comparing the performance of the methods described in Chapters 2 and 5. The first is the projection method and the second is the interior-point primal-dual path-following method employing the NT-direction. The latter was used to solve five different formulations of Problem (1.3.1).(see Sections 3.3.2, 3.3.3, 4.3.1, 4.3.2 and 4.3.3)

A Matlab code was written to implement the projection method. The iteration is stopped when $\|P_S(P_H(F_j)) - P_H(F_j)\|_F \leq 10^{-8}$.

For the other methods, the software SDPT3 ver. 3.0 [46, 44] was used because of its numerical stability [18] and its ability to exploit sparsity very efficiently. The default starting iterates in SDPT3 were used throughout with the NT-direction. The choice of the NT-direction came after some preliminary numerical results. The other direction is HKM-direction which we found less accurate, although, faster than the NT-direction. However, the difference between the two in speed is not of significant importance.

The problem was converted into the five formulations described in Chapters 3 and 4. A Matlab code was written for each formulation. This code formulates the problem and passes it through to SDPT3 for a first time. A second run is

done with the optimal iterate from the first run being the initial point. This process is repeated until no progress is detected. This is done when the relative gap (see [47, 44]):

$$\frac{|P - D|}{\max\{1, (|P| + |D|)/2\}}$$

of the current run is the same as the preceding one. (Here, P and D denote the optimal and the dual objective values, respectively.)

Before we start our experiments we should justify why we exclude the SDV formulation (3.3.6). For this purpose, we test this formulation by examining its performance, in terms of giving accurate solutions and in terms of time and number of iterations, against the projection method. Table 6.1 shows that SDV can not cope with the projection method in all aspects. SDV needed 1003 seconds (almost seven minutes) to reach the solution with relatively small matrix ($n = 30$), furthermore the machine we use to do these experiments could not handle the case when $n = 50$ due to run out of memory. This remark is enough to give up trying to use this formulation. The reason behind this poor performance is, as we said earlier, the size of the SDP problem, which gets larger rapidly as the data matrix get larger and this affects the amount of work needed at each iteration, $\mathcal{O}(n^6)$.

Size	Projection			SDV		
	Time	Iterations	Norm	Time	Iterations	Norm
10	0.6	383	89.1727	3	18	89.1727
30	18	1823	293.2931	1003	40	293.2931

Table 6.1: Performance comparison between SDV and the projection method.

Our numerical experiments were carried out on eleven randomly generated square matrices with different sizes, namely: 10, 30, 50, 100 and 200, two for each size and one of size 400. Each matrix is dense and its entries vary between -100 and 100 exclusive.

All numerical experiments in this chapter were executed in Matlab 6.1 on a 1.7GHz Pentium IV PC with 256 MB memory running MS-Windows 2000 Professional.

Size	Time (sec.)					
	Pro.	SDH	SDQ	SQH	SQQ	SQV
10	2	2	1	1	1	1
	9	1	1	1	1	1
30	11	5	4	3	4	2
	14	5	4	2	2	2
50	117	10	12	5	7	5
	30	11	11	4	3	5
100	61	53	64	28	20	28
	1003	48	42	22	25	21
200	16239	389	284	324	322	284
	4883	355	420	255	268	230
400	36556	4970	3913	3775	4098	2505

Table 6.2: Performance comparison (time) among the projection method and the path-following method with the formulations SDH, SDQ, SQH, SQQ and SQV.

Table 6.2 compares the CPU time. We notice that the consumed time gets larger more rapidly in the projection method with the size of the data matrix F. An obvious remark is that the projection method is the slowest; indeed, it is at least seven times slower than the slowest of the five formulations of the path-following method. However, the difference in time between the five formulations is not big enough to have a significant importance.

Size	Iterations					
	Pro.	SDH	SDQ	SQH	SQQ	SQV
10	1253	16	18	14	14	11
	6629	18	17	14	14	11
30	1215	34	32	35	47	24
	1443	33	33	29	29	20
50	4849	32	41	25	36	24
	1295	32	42	22	18	26
100	504	34	45	27	19	26
	8310	33	28	23	26	20
200	22672	31	22	33	31	25
	6592	28	32	23	27	22
400	7870	28	25	26	26	18

Table 6.3: Performance comparison (number of iterations) among the projection method and the path-following method with the formulations SDH, SDQ, SQH, SQQ and SQV.

Another clear advantage is in terms of number of iterations as shown in Table 6.3. Although the amount of work in each iteration is different for each method, it is still fair to consider it to be a comparison factor.

Table 6.4 shows how close, in Frobenius norm, the optimal solution of each method, H^*, to the data matrix F. The projection and the path-following methods with the formulation SDH, SQH and SQQ gave the same result to some extent. The formulation SDQ couldn't cope with the others as the problem size gets larger. The poor performance of this formulation is due to the matrix P being of full rank. The formulation SQV is less accurate than SDH, SQH and SQQ which is reasonable especially if we notice that the dimension of the second-order cone in this formulation is $1 + n^2$. (see Table 4.1)

Size	Norm					
	Pro.	SDH	SDQ	SQH	SQQ	SQV
10	96.6226	96.6226	96.6226	96.6226	96.6226	96.6226
	94.8320	94.8320	94.8320	94.8320	94.8320	94.8320
30	307.9339	307.9339	307.9406	307.9339	307.9339	307.9339
	327.6784	327.6784	327.6784	327.6784	327.6784	327.6784
50	494.3805	494.3805	494.5038	494.3805	494.3805	494.3805
	497.4383	497.4383	497.6330	497.4383	497.4383	497.4383
100	991.8832	991.8832	994.8612	991.8832	991.8832	991.8833
	997.4993	997.4993	998.8048	997.4993	997.4993	997.4994
200	1986.9397	1986.9398	1990.0924	1986.9402	1986.9402	1986.9414
	1994.8409	1994.8410	1998.6048	1994.8410	1994.8410	1994.8418
400	3998.4967	3998.5047	4001.9242	3998.5007	3998.5007	3998.6166

Table 6.4: Performance comparison (norm $\|H^* - F\|_F$) among the projection method and the path-following method with the formulations SDH, SDQ, SQH, SQQ and SQV.

To summarize the above discussion, we introduce Table 6.5. This table gives a measure to how close the optimal solutions of SDH, SDQ, SQH, SQQ and SQV from that of the projection method which is the most accurate. The error is computed simply by evaluating the difference between the norm $\|H^* - F\|_F$ of the projection method and the norm obtained by the different formulations of the path-following method. This table shows that formulations SQH and SQQ give almost the same results; I think this is due to that the second-order part of both formulations is the same except some entries are different by a constant throughout the problem.

We conclude this chapter by addressing two remarks. One is that the projection method, despite its accuracy, is very slow. Whereas, the path-following

Size	Error				
	SDH	SDQ	SQH	SQQ	SQV
10	6.3×10^{-9} 6.4×10^{-9}	3.4×10^{-9} 3.2×10^{-8}	6.1×10^{-9} 3.6×10^{-8}	6.1×10^{-9} 3.6×10^{-8}	1.3×10^{-5} 1.2×10^{-5}
30	7.5×10^{-10} 1.6×10^{-9}	6.7×10^{-3} 9.0×10^{-9}	2.6×10^{-8} 2.0×10^{-9}	3.0×10^{-8} 2.0×10^{-9}	9.7×10^{-8} 1.2×10^{-8}
50	1.9×10^{-9} 3.7×10^{-9}	1.2×10^{-1} 0.2	8.9×10^{-9} 7.8×10^{-9}	9.0×10^{-9} 8.0×10^{-9}	2.1×10^{-5} 2.1×10^{-5}
100	5.1×10^{-10} 9.2×10^{-10}	3.0 1.3	1.8×10^{-8} 5.8×10^{-8}	1.8×10^{-8} 5.8×10^{-8}	1.0×10^{-4} 1.5×10^{-4}
200	6.6×10^{-5} 1.1×10^{-4}	3.2 3.8	4.4×10^{-4} 9.1×10^{-5}	4.2×10^{-4} 9.1×10^{-5}	1.6×10^{-3} 9.3×10^{-4}
400	8.0×10^{-3}	3.4	4.0×10^{-3}	4.0×10^{-3}	1.2×10^{-1}

Table 6.5: Performance comparison (error)

method with SDH, SQH and SQQ formulations is very fast, sometimes more than 40 times faster than the projection method (see Table 6.2 when $n = 200$), and gives results of acceptable accuracy. The other is that we did not gain any considerable advantage out of solving our problem as a mixed semidefinite and second-order cone problem (SQH, SQQ and SQV). This can be seen clearly by noticing the good performance of the formulation SDH, which solves the problem as an SDP problem.

Chapter 7

Conclusions and Further Research

In this thesis we have studied the Hankel matrix approximation problem (1.3.1) from a new point of view. We looked at it as an SDP problem and then as a mixed problem of semidefinite and second-order cone constraints. We gave different formulations, namely: SDV, SDQ, SDH, SQV, SQQ and SQH. Then we solved it using the primal-dual path-following method. We found that it is no longer efficient to solve Problem (1.3.1) using the projection method. We found also that the way of formulation does have an effect on the performance of the primal-dual path-following method. The formulations SDH, SQQ and SQH seem to be the best among the others.

What we have achieved in the work suggests some further investigations. These suggestions may be summarized as follows:

In the projection method, at each iteration we find the spectral decomposition of an $n \times n$ matrix which requires floating point operations of $\mathcal{O}(n^3)$. This step consumes almost 80% of the total cpu-time. Recently, Luk et. al. [31] proposed a new algorithm that can find all the eigenvalues of an $n \times n$ Hankel matrix in $\mathcal{O}(n^2 \log n)$ operations. Thus, one can try to use

this method to find the projection of a Hankel matrix on the cone of positive semidefinite matrices. However, more work on the implementational aspects of this algorithm is needed.

Related to the above point, a further investigation should be made in order to see how efficient the projection is when the eigenvalues are estimated instead of being exactly calculated. Especially, if we know that the estimating process using Lanczos iteration requires only $\mathcal{O}(n^2)$ operations.

The approach we follow in this thesis may be used to deal with similar structured problems such as the problem of finding the nearest symmetric positive semidefinite Toeplitz matrix to a given data matrix. In this context, we introduce the following vectorization operator, **tvec**, (similar to **hvec**), which may be useful in the formulation process.

$$\mathbf{tvec}(T) = [\sqrt{n}t_1, \sqrt{2(n-1)}t_2, \sqrt{2(n-2)}t_3, \cdots, \sqrt{2}t_n]^T,$$

where T is an $n \times n$ real symmetric Toeplitz matrix, *i. e.* , T has the following structure:

$$T = \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ t_2 & t_1 & \cdots & t_{n-1} \\ \vdots & \vdots & \ddots & t_2 \\ t_n & t_{n-1} & \cdots & t_1 \end{bmatrix}.$$

Another example which may be studied is a problem with Hankel-plus-Toeplitz structure.

Another version of Problem (1.3.1) which requires the optimal solution to

be of pre specified rank. This problem was solved using hybrid methods [1] composed of projection and SQP methods. We think that this problem may be solved more efficiently using convex optimization, although, the constraint concerned with the rank is neither smooth nor convex. This problem needs further investigation.

Finally, we suggest more work on the development of a special version of the path-following method to deal with our problem and other structured problems in a unified manner to give better results.

References

[1] S. Al-Homidan. Combined methods for apprpximating Hankel matrix. *WSEAS Transactions on systems*, 1:35–41, 2002.

[2] S. Al-Homidan. Hybrid methods for approximating Hankel matrix. *Numerical Algorithms*, 32:57–66, 2003.

[3] F. Alizadeh. The semidefinite programming page. http://rutcor.rutgers.edu/ alizadeh/sdp.html.

[4] F. Alizadeh. Optimization over the positive-definite cone: Interior point methods and combinatorial applications. In P. Pardalos, editor, *Advances in Optimization and Parallel Computing*. North-Holland, the Netherlands, 1992.

[5] F. Alizadeh and D. Goldfarb. Second-order cone programming. Technical Report RRR Report 51-2001, RUTCOR, Rutgers University. 6, 2001.

[6] F. Alizadeh, J. A. Haeberly, M. V. Nayakkanakuppann, M. Overton, and S. Schmieta. SDPPack, user's guide, 1997.

[7] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM J. Optim.*, 8:746–768, 1998.

[8] J. Allwright. Positive semidefinite matrices: Characterisation via conical hulls and least-squares solution of a matrix equation. *SIAM J. Control Optim.*, 26:537–556, 1988.

[9] R. Bellman and K. Fan. On systems of linear inequalities in Hermitian matrix variables. In V. L. Klee, editor, *Convexity*, volume 7, pages 1–11. Proc. Symposia in Pure Mathematics, Amer. Math. Soc., Providence,RI, 1963.

[10] J. P. Boyle and R. L. Dykstra. A method of finding projections onto the intersection of convex sets in Hilbert space. *Lecture Notes in Statistics*, 37:28–47, 1986.

[11] J.P. Burg, D. G. Luenberger, and D. L. Wenger. Estimation of structured covariance matrices. *Proc. IEEE*, 70:963–974, 1982.

[12] B. Craven and B. Mond. Linear programming with matrix variables. *Linear Algebra with Appl.*, 38:73–80, 1981.

[13] E. de Klerk. *Aspects of Semidefinite Programming: Interior Point Algorithms and Selected Applications*, volume 65 of *Applied Optimization Series*. Kluwer Academic Publishers, 2002.

[14] F. Deutsch. Von neumann's alternating method: the rate of convergence. In C. Chui, L. Schumaker, and J. Ward, editors, *Approximation Theory IV*, pages 427–434. Academic Press, New York-London, 1983.

[15] R. L. Dykstra. An algorithm for restricted least squares regression. *J. Amer. Stat.*, 78:839–842, 1983.

[16] W. Fang and A.E. Yagle. Two methods of Toeplitz-plus-Hankel approximation to a data covariance matrix. *IEEE Trans. Signal Processing*, 40:1490–1498, 1992.

[17] R. Fletcher. Semidefinite constraints in optimization. *SIAM J. Control Optim.*, 23:493–513, 1985.

[18] K. Fujisawa, M. Fukuda, M. Kojima, and K. Nakata. Numerical evaluation of SDPA (semidefinite programming algorithm). In H. Frenk, K. Roos, T. Terlakey, and S. Zhang, editors, *High Performance Optimization*, pages 267–301. Kluwer Academic Press, 2000.

[19] W. Glunt, L. Hayden, S. Hong, and L. Wells. An alternating projection algorithm for computing the nearest Euclidean distance matrix. *SIAM J. Matrix Anal. Appl.*, 11(4):589–600, 1990.

[20] K. M. Grigoriadis, A. E. Frazho, and R. E. Skelton. Application of alternating convex projection methods for computing of positive Toeplitz matrices. *IEEE Trans. Signal Processing*, 42:1873–1875, 1994.

[21] S. P. Han. A successive projection method. *Math. Programming*, 40:1–14, 1988.

[22] C. Helmberg. Semidefinite programming home page. http://www.zib.de/helmberg/semidef.html.

[23] C. Helmberg, F. Rendl, and R. Vanderbei H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM J. Optim.*, 6:342–361, 1996.

[24] K. Hoffman and R. Kunze. *Linear Algebra*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1971.

[25] I. S. Iohvidov. Hankel and Toeplitz matrices and forms, algebraic theorey. Translated by G. Philip A. Thijsse.

[26] N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[27] L. Khachiyan. A ploynomial time algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[28] M. Kojima, S. Kojima, and S. Hara. Linear algebra for semidefinite programming. Technical report, Dept. Mathematical and Computing Sciences, Tokyo Institute of Technology,Oh-Okayama,Meguro-ku,Tokyo 152, Japan,, October 1994.

[29] M. Kojima, S. Shindoh, and S. Hara. Interior-point methods for monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM J. Optim.*, 7:86–125, 1997.

[30] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and Applications*, (284):193–228, 1998.

[31] F. T. Luk and S. Qiao. A fast eigenvalue algorithm for Hankel matrices. *Linear Algebra and its Applications*, 316(1–3):171–182, 2000.

[32] C. S. Macinnes. The solution to a structured matrix approximation problem using Grassman coordinates. *SIAM J. Matrix Anal. Appl.*, 211(2):446–453, 1999.

[33] R. D. C. Monteiro. Primal-dual path-following algorithms for semidefinite programming. *SIAM J. Optim.*, 7:663–678, 1997.

[34] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Time Methods in Convex Programming*. SIAM, Philadelphia, 1994.

[35] Yu. E. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Math. Oper. Res.*, 22:1–42, 1997.

[36] Yu. E. Nesterov and M. J. Todd. Primal-dual interior-point methods for self-scaled cones. *SIAM J. Optim.*, 8:324–364, 1998.

[37] J. Von Neumann. *Functional Operators II, The geometry of orthogonal spaces*. Annals of Math. studies No.22, Princeton Univ. Press., 1950.

[38] J. Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Math. Programming*, 40(1):59–93, 1988.

[39] A. Shapiro. External problems on the set of nonnegative definite matrices. *Linear Algebra with Appl.*, 67:7–18, 1985.

[40] A. K. Shaw, S. Pokala, and R. Kumaresan. Toeplitz and Hankel matrix approximation using structured approach.

[41] J. Sturm. Implementation of interior point methods for mixed semidefinite and second order cone optimization problems. Technical report, August 2002.

[42] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[43] Y. J. Suffridge and T. L. Hayden. Approximation by a Hermitian positive semidefinite Toeplitz matrix. *SIAM J. Matrix Analysis and Appl.*, 14:721–734, 1993.

[44] R. Tütüncü, K. Toh, and M. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. to appear.

[45] M. J. Todd. Semidefinite Optimization. *Acta Numerica*, 10:515–560, 2001.

[46] M. J. Todd, K. C. Toh, and R. H. Tütüncü. On the Nesterov-Todd direction in semidefinite programming. *SIAM J. Optim.*, 8:769–796, 1998.

[47] M. J. Todd, K. C. Toh, and R. H. Tütüncü. SDPT3 — a Matlab software package for semidefinite programming. *Optim. Methods Softw.*, 11:545–581, 1999.

[48] K. C. Toh. A note on the calculation of step-lengths in interior-point methods for semidefinite programming. to appear.

[49] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Rev.*, 38, 1996.

[50] N. Wiener. On factorization of matrices. *Comm. Math. Helv.*, 29:97–111, 1955.

[51] H. Wolkowicz. Some applications of optimization in matrix theory. *Linear Algebra with Appl.*, 40:101–118, 1981.

[52] H. Wolkowicz, R. Saigal, and L. Vandenberghe. *Handbook of Semidefinite Programming*. Kluwer Academic Publishers Group, Boston-Dordrecht-London, 2000.

[53] S. J. Wright. Interior-point methods online homepage. http://www-unix.mcs.anl.gov/otc/InteriorPoint/.

Vita

Mohammed Mogib Alshahrani,
Graduated from Abha's Teachers' College, Math major, second semester 1997.
E-mail: mmogib@awalnet.net.sa